Service Oriented Code Generator for Fast Prototyping

Based on Requirement Definition Schema

By D.U.I.Hewage

149211G

Faculty of Information Technology

University of Moratuwa

May 2017

Service Oriented Code Generator for Fast Prototyping

Based on Requirement Definition Schema

Software tool providing easy prototyping ability for web applications

D.U.I. Hewage

149211G

(MSCIT/14/028)

Dissertation submitted to the Faculty of Information Technology, University of Moratuwa, Sri Lanka for the partial fulfillment of the requirements of the Degree of MSc in Information Technology.

Faculty of Information Technology

University of Moratuwa

May 2017

Declaration

I declare that this thesis is my own work and has not been submitted in any form for another Masters, Degree or diploma at any university or other institution of tertiary education. Information derived from the published or unpublished work of others has been acknowledged in the text and a list of references is given.

Name of the Student:	Signature of the Student
D. U. I. Hewage	
	…………………………..
	Date:
Supervised by:	Signature of the Supervisor
Mr.Chaman Wijesiriwardana	……………………………
	Date:

Dedication

This dissertation is dedicated to my beloved parents, siblings, nimshi and her parents, my teachers who gave me endless courage and support to achieve my task and goal in completing the research project.

Acknowledgement

My heartiest thanks go to my supervisor Mr. Chaman Wijesiriwardana for the guidance, assistance, encouragement, valuable advices on improving the research and providing this opportunity carry out this research project.

Also, sincerely thanks to all my teachers who taught in MSc IT degree program. Things leant from these subjects made it easier to make this research project a successful one.

Last but not least, a sincere thank goes to everyone who supported specially teams from Redot Pvt Ltd and Effro Pte Ltd Singapore for contributing their valuable time on this research.

Abstract

With the rise of the latest web technologies, it has now become the mostly used software solution for lots of business areas. Because of its flexibility and easy connections between the clients and the server made via clouds, it's the most popular technology solution provider for the industry. Having said that, new and faster approaches for programming, planning, deploying and testing are being introduced at a rapid speed. As a result of this, large number of new tools and technologies are popping up in the industry. Even though they have introduced these tools and technologies to ease up development work, still there are some areas which are time consuming and costly for the management. To be specific Database Designing, Initial Project setup, Authentication module coding, Coding the User Interfaces, Writing CRUD functions covering every use case of the applications, setting up deploying mechanism, writing test cases, and many more tasks are still done manually or taken from a previous project. Even if they have taken it from a previous project they still have to code the CRUD operations and some other things which are not automated yet.

Goal of this research is to find out a proper automating mechanism for most of the tasks which are not yet automated using available open source projects and to combine a set of task specific automating tools to act as a complete solution. Although there are some systems available which developers are using to reduce repetitive work, and can manage their work using these systems, it can be improved further and save days to weeks from their development time. This way it can avoid concerns over the cost involved with the implementation because of the time spent on these repetitive tasks.

This project proposes a customized solution for avoiding repetitive tasks in software implementation. Reducing the time spent on these tasks is the main objective of this project which ultimately leads to a software prototyping application. Since the modern approach of software implementation is model driven, proposed system will also consist the model driven approach with its solution. The proposed system is capable of source code generation. Pre-generated source code can reduce the time spent on coding, use a model driven approach, automate validations, maintain coding standards as well as the generate UI elements which are required to display data on frontend. Using only 2 schema files written in JSON format which describes the flow of the application, models and validation systems are capable of understanding and generating code based on these definition files. Since the application is using a very high level of definition of the user requirement, it is a forward engineering approach. Keeping in mind of the latest technologies and the mobile technology evolvement, the generated code will consist of 2 sections, namely the Front-end and the Back-end. Front end consists of the UI information and the flow of the application which the final client or the customer will experience. Back end is consisting of a highly customizable REST API which supports mobile implementation as well. To ease up the implementation of this project, project is using set of open-source projects such as angular-seed, expressjs. And the solution is provided using NodeJs. Model-Driven Application Prototyping and Code Generation using Forward Engineering System (FES) supports code generating in multiple languages and supports multiple DBMSs such as MySql, SQL, MongoDB, etc. I have tested the system with the collogues at my work place which all of them are developers. And currently using the system for generating first and second level prototypes.

Finally, I have achieved the task of implementing a software solution which generates model driven, initial project setup, reusable and testable code, supports multiple databases and greatly reduce time spent on repetitive work. And in the first phase of the most web application projects in can reduce the project setup time by nearly 1.5 weeks as well as 4-10 hours of 1 module of the code.

Keywords: prototyping, model driven, code generation, forward engineering, CRUD automation, multi-language

Table of Contents

Table of Tables

Table of Figures

Chapter 1

1. Introduction

1.1. Prolegomena

Modern world has already been taken over by technology, as it has come to a point where it is not even possible to live without it. Before 20 years ago, no one has ever imagined that technology would become so advanced that technology would play a major role in areas such as food processing, clothing, medical research, water management, electricity plants, etc. Considering the complex day to day life of a human-being and the rapid growth of the world population, it won’t even be able to cater for basic human needs without the modern-day technology. While technology is playing a major role in delivering basic human needs, it has also become the ultimate solution provider for economics, transportation, sales and marketing, banking and lots of other areas.

As technology started connecting people all around the world with the innovation of the world wide web, websites and web pages were introduced just for delivering information. It was initially used for data communication and very rarely for marketing. When the technology started evolving and becoming more popular, WWW has started to provide vast variety of solutions to people around the world such as sales and marketing, entertainment, streaming technologies, mailing services, etc. As the requirement for these services increased rapidly, software development companies began getting busier and overloaded with projects. For catering to a large number of projects and requirements, development speed of the software products had to be increased respectively. Catering to a large number of projects and requirement means a larger client base and increased profit for a company.

As a result, addressing a large number of requirements at the same time has become a problem and was very difficult to maintain. Developers and engineers were keener on experimenting and researching on new things which will make sure each phase of software development methodology will take less time while maintaining quality which will benefit for both the client and the company which provides the software solutions.

For this purpose, people have introduced a lot of various software development methodologies such as Waterfall, Prototyping, Agile, etc. As an example, SRS documents, Gantt charts, design diagrams, client meeting documents, etc were introduced for the planning phase. Modern-day software development uses different methods and tools during each phase of the development life cycle. Specifically, for implementation phase, they have introduced different programming languages such as PHP, C#, Java, JavaScript, Python, Ruby, etc. For each of these languages they have introduced different frameworks, design patterns such as OOP, MVC [1], Singleton, Model-Driven architecture, Modularized architecture, etc.

Speeding up the development process, improved product quality, improved security, highly reliable and maintainable code, satisfied clients, satisfied project teams and increased profits are some benefits of using these technologies. Even though these tools and methods increased profits and productivity of the software tool it has made the programing language more complex and experienced personnel only found them useful. As novice developers are spending more time on researching and learning on each of these frameworks and best practices, this makes it more difficult to novice developers who come on board as a team member to catch up with others. Teaching them these technologies requires more time and effort.

As a result, industry experts are now researching on code generators which is capable of generating software source code based on a requirement. These generators try to reduce time spent on repetitive tasks, generate source code with industry accepted standards, follow widely used design patterns and frameworks.

This research is an effort on implementing a software tool which is capable of generating a working prototype source code based on a given requirement and researching on a sophisticated method defining client requirement which is understandable by a computer. Solution is supposed to address set issues faced by developers and other project members on day to day basis which we will be discussed further during the problem domain section.

1.2. Background and Motivation

When it comes to software development, there are set of things which team members do repeatedly. Specifically, teams do lots of client meetings trying to clear up requirements, system architectures work hard on designing the best possible solution based on client requirement and this includes large number of iterations and fine tuning. When the development team initiate the implementation, they are involved in ground up work which is almost similar for most of the projects. This includes setting up version controlling, setting up software frameworks, initiating dependency managers, setting up testing and continuous integration tools, setting up deployment tools. Modern-day trend is that the programmers divide software solution in to modules and complete one by one. These modules use different frameworks and programming practices. Most of the time, final operation of these modules is involved in similar tasks such as create, read, update, delete which is CRUD [1] regardless of what the business logic is. Once the implementation is over it is then passed to the testers and they are responsible in validating the software product. Testing also involves testing CRUD operations mentioned above which are similar for most of the modules.

These repetitive tasks can be categorized into two sections. Namely short term repetitive tasks and long term repetitive tasks. Client meetings for requirement clarifications, System design fine tuning and CRUD operations can be identified as short term repetitive tasks. Ground up work for each project can be identified as a long term repetitive task.

With the emerge of web development and software engineering technologies, various methods and tools were introduced addressing these repetitive tasks. Specifically, these tools included CRUD automation, user interface generation, systems which do ground up work for developers, systems which can generate the flow of a system, etc.

Webratio [2] is a product which was designed to generate the system flow and the source code of it. This product is capable of generating source code in Java and it provided interactive UI which can be used to design the system. PHP grocery CRUD [3] is a work which can generate CRUD operations for Mysql database tables. yeoman generator [4] is a collection of source code generators in which developers can use to do ground up work for them. This tool helps to avoid long term repetitive tasks and speed up the initial project setup time. And the industry experts report that they have gained significant improvement in productivity by introducing these tools to their development process.

Similarly, With the advancement of other programming frameworks such as laravel [5] and expressJs for Node [6] which follows model-driven MVC architecture, implementation of Web APIs and Web applications have become much easier than it was previously. Experts suggests maintainability improvements with respect to the code-centric implementation, improved efficiency, improved effectiveness and optimization of development effort using these tools.

Even though these tools were capable of reducing the time taken for repetitive tasks by generating source code they were generating a generalized code which can then be modified to work with other modules as well. To overcome this issue there should be a mechanism to feed the requirement into the system and generate a customized code for each of these modules which will reduce the time taken further. Research done in Ibn Tofail University, Kenitra [7] the authors presented an approach based on model transformations that automatically generates the CRUD operations for a web system taking class diagrams based on UML (Unified Modeling Language) profiles as an input. Research done at university of twente Netherlands [8] the authors also evaluated the productivity improvements obtained by a model-driven code generation approach which automatically generates the CRUD operations for a Web information system.

This approach also takes UML class diagrams as an input. By using these generators, the authors observed an important development time reduction (up to 90.98%). They also surveyed the developers about the difficulties found compared to the manual coding approach and obtained better results for the code generation approach.

A research done at the Department of Engineering University of Sannio, Italy [9] suggests a class definition using XML language which will then be used to generate the code. The process and the technologies adopted to implement this approach can be reused to develop the fast prototyping approach for a different design model and/or a different target technology platform. Another research done at University of Lisbon, Portugal [10] also suggest a model definition using XML language and generation CRUD operations using MVC design patterns.

Even though these tools can play handy in different scenarios, there set of issues which hasn't been addressed yet. These will be discussed in the next section.

1.3. Problem domain

Information technologies represent one of the fastest developing business areas [10]. Specifically, Web applications are usually required to be developed and delivered in a very short period of time and after that to be updated and evolved even faster [9]. Reduce time to market, reduce the cost of development (that depends on time), standardize software development, improve quality, improve reliability and reduce the complexity in process management [11] have become the crucial factors for getting a job done and getting it delivered. This very short development life cycle often forces developers to focus more on implementation and devote low effort and a short time to the design phase which in the end negatively affects the quality of the web application. The question is how these costs can be reduced thus improve productivity and surpass competition. As discussed previously on the issues faced by modern day developers as well as the solutions and researches done to overcome these issues, they are still not capable of solving some of the major scenarios. Namely,

CRUD automation

Modern CRUD automation systems are capable of generating high performance source codes. But most of them are not capable in generating source code which can manage data integrity and relationships between database tables or models. Source code which was generated using these tools should be re-modified to work with the existing project codebase. At the same time, these tools cannot be used as a starting point for a project as the initial development environment or ground up work should be done before using.

Seed project generators

As discussed before the yeoman generator is a tool which can be used to generate a seed project. This tool is useful when a new project begins as it is taking care of doing ground up work such as setting up version controlling, initial project code, setting up testing framework etc. after using for initiation of the project it has no use. It is not capable of generating any CRUD operation neither the application flow.

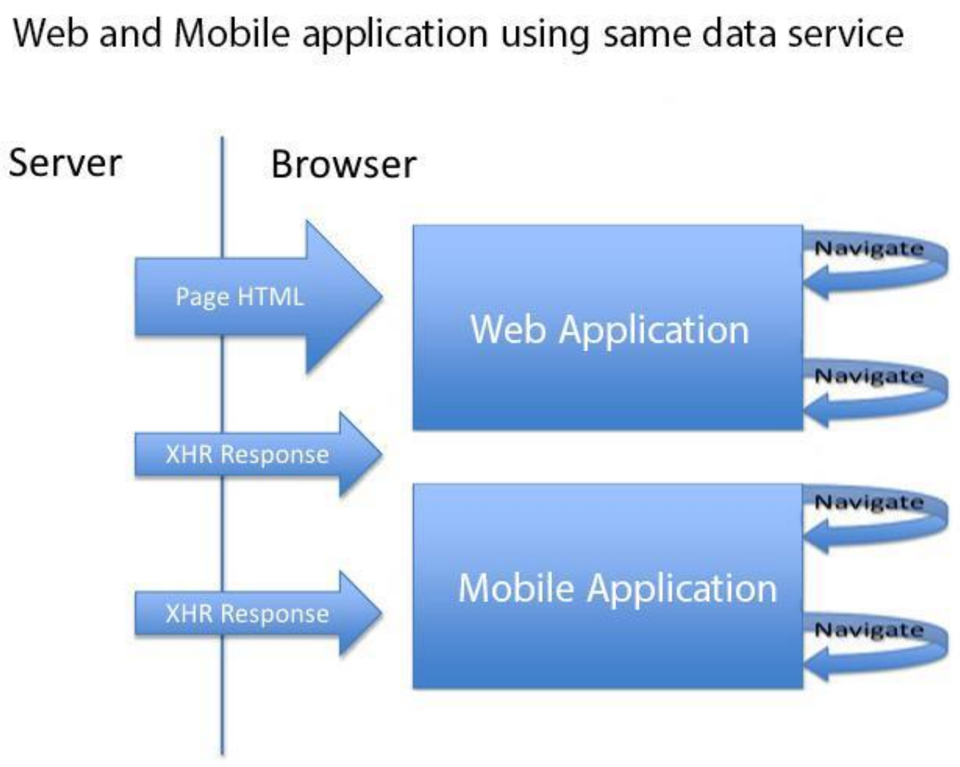
Requirement definition language

Some researchers suggest that using requirement definition, they are capable of generating source code for a given requirement. Although using requirement definition mechanism is useful, most of these tools takes an UML diagram or a class definition as an input. Drawback on this approach is that developers or designers has to spend time on designing either UML or Classes before using these tools.

Modern technology support

Modern day web applications run on various devices, browsers and platforms. These includes mobile applications as well. Usually mobile applications are powered by data services such as REST API [12] or SOAP services [13]. None of the above-mentioned applications are capable of generating a good data service. According to modern-day practices there should be a good separation between data layer and the presentation layer. By having a separate data layer makes it possible to use the data service for other applications such as mobile as standalone applications.

A good research on filling this gap should be conducted to overcome these issues faced by the project teams.



1.4. Hypothesis

By conducting an extensive research on the issues faced by the project teams, we have identified set of issues that need to be addressed. In this regards our hypothesis is that, **an implementation of a software tool which can take in a unique customer requirement as an input defined in a requirement definition file and output a working prototype source code.**

Generated software tool should be able to use easily and should be available anywhere since the goal is to reduce the time taken for repetitive tasks. Because of this the proposed solution will be delivered in two variations one as a service and where anyone can access as they want. And one as a Real-Time application where it watches for requirement changes in the requirement definition file and build source code on the fly. Project is titled according to this hypothesis,

Service Oriented Code Generator for Fast Prototyping using Schema Based Requirement Definitions

1.4.1. Expected Features

- Automate the CRUD operations
- Should complete the ground up work mentioned
- Generate the flow of the required system as well as the user interfaces
- Separation between data layer and the presentation layer
- Highly Model-Driven
- Generate the source code which follows industry standards with MVC design pattern

1.5. Aim and Objectives

1.5.1. Aim

Aim of this research is to reduce the time taken on day to day repetitive tasks while maintaining quality of the project. This leads to happy clients, happy project teams as well as increased profits.

1.5.2. Objectives

Main objective of this research is to implement a software tool which will generate a prototype application source code using the provided set of requirement definition schema files.

A good requirement definition mechanism should also be implemented which is should be able to define the flow and different states of the system, data models and relationships between them, UI elements, user input validation rules and output messages.

1.6. Structure of the Thesis

The best of the thesis is organized as follows. Chapter 1 critically reviews the literature on current prototyping tools and source code generation tools such as CRUD tools. Chapter 2 is discussing about the current developments. As well as the issues and technologies different people/ organizations have used to overcome the issues with prototyping and automated code generation. Chapter 3 is about the technologies used in implementing the solution. Chapter 4 presents new approach to build fully functional prototypes using schema definitions. Chapter 5 and 6 describe the design and implementation respectively. Chapter 7 is on the evaluation of the solution and it discuss of 2 case studies conducted. Chapter 8 concludes the research with a note on further work.

1.7. Summery

This chapter described the overall description of the research and introduced the research problem and the solution. Next chapter is the literature review which will discuss the work of other researchers on the same domain. it will provide full detailed information about background information of the project based on a literature survey as well as it’ll provide information regarding the current development challenges.

Chapter 2

2. Current Development and Challenges

2.1. Introduction

Chapter 1 gave a comprehensive description of the overall project described in this thesis. This chapter provides a critical review of the literature in relation to developments and challenges in using prototyping tools and automated code generation. For this purpose, the review of the past researches, software's and articles have been presented under three major sections. Namely, early developments, modern trends and future challenges. At the end, this chapter defines the research problem.

2.2. Current developments

2.2.1. Toward automatic generation of mvc 2 web applications

This paper [1] is more focused on Model Driven Architecture (MDA) for automating the code generation. And it discusses the advantage of using the MDA. According to the conclusion,

"They have applied the MDA approach in web applications engineering. This particularly generates the makings of a web application on the basis of a UML class diagram. The latter is built on the basis of different attributes of the information system. The generation process will provide an opportunity for the user to add, edit, delete, and especially display the various objects he needs. He must be able to display objects of a given class, based on information from another object of another class provided the two classes are connected via associations using a class diagram. To achieve this, they first develop the source metamodel managing UML class diagrams. At the target metamodel, they have developed all meta-classes needed to be able to generate an application respecting a MVC2 architecture. The mapping rules were developed and put together in a transformation algorithm. This algorithm makes it possible to browse the source class diagram and generate through these rules, an XML file containing all actions, forms, and then forwards jsp pages that can be used to generate the necessary code of the target application. This is very useful when dealing with related information between them in a tree structure and the display of information depends on another. This work can be extended to support advanced aspects of the content of Web pages to produce a web application from start to finish, i.e. providing the user's interface part at a will and appropriate treatment responding to requests. In perspective, this work should be extended to allow the generation, in addition to the configuration files, of other components of the Web application: model, view, controller and their constituents. Emphasis should be placed on the support of other CRUD methods such as create, remove and update. "

Advantages

Software tool which was built by this research is capable of understanding UML and class diagrams and generate the required product based on the input. The code which gets generates follows mvc 2 architecture and model-driven approach.

2.2.2. A Generator of MVC-based Web Applications

This research [2] is also based on generating mvc based web application. as extracted from the research paper,

The generator presented is developed cater customized source code generation. Based on a practical experience in designing generator for a commercial application, it presents a possibility for fast development of a custom generator as a solution for a specific problem. The generator relies on the Hibernate Tools toolset which is responsible for a meta model creation. Thus, most of the time was spent on designing templates which implement specific technologies and structure of an application. The main characteristics of the presented generator are simplicity of its design, short time spent on its development and full suitability to application specific requirements. Despite its simplicity, the generator allows some customizations and controls of the code generation process. It is primarily enabled by using functionality of the hibernate.reveng.xml file. Testing the generator on The Asset Management System database which **contains 400 tables, 200 functionalities based on 200 dictionary tables** were fully generated without need for additional customization. Coding these functionalities manually would **require approximately 800 hours**, assuming that 4 hours are required for coding and testing one simple functionality. Since the enterprise applications have (or should have) more or less uniform code structure and visual appearance, the generator was able to produce basis for more complex functionalities, which were available for later easy upgrading. Some of the specific functionalities such as login page and supporting program logic could not be produced by the generator and had to be developed manually.

Advantages

This software tool which was the outcome of the research was also taking few xml files as an input and outputs a java based web application. uses java hibernate framework as the programming framework.

2.2.3. A Model-Driven Approach for the Fast Prototyping of Web Applications

This paper [3] presented an approach for the model-driven fast prototyping of Web applications developed using Eclipse technologies and frameworks such as EMF, GMF and Xpand 2. The approach consists of a two-step process, modeling-generate, and is accompanied by two supporting tools. a modeling tool for defining the design of the application by adopting the Model-View-Controller architectural design pattern, and a generator tool that transforms the defined design model into a “ready to run” prototype of the application. The code generation phase is fully automated and produces an Eclipse Faceted Web dynamic project that uses the J2EE JavaServer Faces implementation technology and is ready to deploy on a Tomcat-MySQL platform. A case study conducted on designing and fast prototyping a Web application for online note taking and sharing has shown that the approach is valid and the supporting tools work properly. In particular, the approach enables effortlessly repeating the development cycle “modeling-generating-validating” to verify and incrementally improve the design of the application. The process and the technologies adopted to implement our approach can be reused to develop the fast prototyping approach for a different design model and/or a different target technology platform.

Advantages

This tool was built as an eclipse plugin. it allows users to design a class diagram and generate the source code for Java. case studies show that it can be used for different scenarios.

2.2.4. XFlash–a web application design framework with model-driven methodology

This research [4] has developed a model-driven methodology for web application development. In terms of web user-interfaces development, they have implemented a generator for generating the user-interfaces components from the XFlash framework. Different from traditional web user-interface development using different computer languages, we implement the web user-interfaces using a single computer language - the ActionScript, and organize the structure of web interfaces using design patterns. Our approach contributes to the reusability of web user-interface components in different applications. this approach generates flash components to implement the web user-interface elements.

Advantages

In xFlash implementation their target is to generate an ActionScript project which is good for Flash based application. same as most of the researches it also uses xml file input to generate a model driven flash application.

2.2.5. Leveraging declarative languages in web application development

This research [5] is based on unified language programming. the presentation tier is expanded to cover all three tiers a Web Application. This Allows end-user developers not only to leverage their existing skills in user interface development, but also to implement entire Web Applications using a single declarative language and data model. This research uses XForms DB framework, all application development is done on the client side. This helps especially Web designers—usually mid-level end-user developers—to become advanced Web Application developers. The framework is based on the XForms markup language and proposed XForms DB server-side language extension. They have implemented the XForms framework based on the derived requirements, and argued that it could simplify both the development and maintenance of small and medium-sized Web Applications.

Advantages

According to researches, this approach improved the time to market as well as it improved the performance of the application. this tool also requires a XML input and outputs a Java based application

2.2.6. An Effective Development Environment Setup for System and Application Software

This is a research [6] conducted on how a development environment should be before starting up a project. the code generator which we use should generate the code with considerable ground up work which makes developers life easier. according to researchers,

The Software Development Environment Model proposed in the paper can significantly improve effectiveness and productivity, and reduce overall costs, as well as improve the quality of the final product. The section "An Effective Software Development

Environment Model Example" offers a combination of standard tools that do not require much time for setup and administration, and naturally fit to the development tasks. Once installed and configured, these tools can work for a long time without needing to be changed.

it also suggests on Automatic build and test procedures, Early defect warning by continuous build and test execution and Automated continuous integration. it is preferred that a code generator follows these guidelines when generating the code.

2.2.7. Other works

Apart from the main researches that we have discussed above, we will look bit more into modern day tools which are being used in the market.

2.2.7.1. Grocery CRUD

Grocery CRUD [7] is an open source library for creating CRUD without any effort of coding. Using a grocery CRUD, you can create a fully successful CRUD system within moments. When this code builder is used, you don't have to rewrite code again and again. It also provides a platform to use common assets like css and js. With a few lines of code the CRUD is ready to be used.

2.2.7.2. Angular-fullstack

AngularJS Full-Stack [8] generator is a yeoman generator for creating MEAN/SEAN stack applications, using ES6, MongoDB/SQL, Express, AngularJS, and Node. It can quickly set up a project following best practices. With the angular-fullstack you can create new endpoints for the server side or client side components (like routes, controllers, services, filters, directives etc. It is easy to install, creates both client and server scaffoldings, introduces good practices in the generated code, server side API prepared to use authentication, support HTML or jade templating on client side, support for different CSS preprocessors and commands to scaffold anything.

2.2.7.3. Webratio

Webratio [9] is a Cloud-based Relationship Management system that allows the user to configure, manage and deliver IoT(Internet of Things)-based business services to the customers, partners and stakeholders. To stand out in the era of Digital Business, you need unique and innovative mobile apps and web applications. It must be developed quickly, often starting with unclear and changing requirements. It provides simple and intuitive development environments that support the lifecycle throughout the applications. This is committed to delivering the best mobile and Web development technologies to allow the users to take full advantage of all the opportunities of the era of Digital Business.

2.3. The Research Gap

To identify the research gap, we evaluated above mentioned researches and its generated source codes under following criteria. these criteria were selected based on the research articles as well as the requirements received from the surveys conducted with industry experts.

1. Code generation support 2. Customized flow generation 3. Model-Driven architecture 4. Customized data model generation 5. MVC design pattern	9. Server-side REST api support 10. Separated front end application 11. Modular source code structure 12. Industry accepted framework usage 13. Multiple language support

6. Relationship mapping and ORM support 7. Real-time code generation 8. JSON support*	14. Customized user input validation support 15. Interactive application designer 16. Cloud based service support

Table 2-1 List of evaluation criteria for existing prototyping tools

*JSON is recommended for simplifying the requirement definition. justification of this technology is discussed in the 5th chapter which is the Design of the system.

Research	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
Toward automatic generation of mvc 2 web applications	✓	x	✓	✓	✓	✓	x	x	x	✓	x	x	x	x	x	x
A Generator of MVC-based Web Applications	✓	x	✓	✓	✓	x	x	x	x	x	x	x	x	x	x	x
A Model-Driven Approach for the Fast Prototyping of Web Applications	✓	x	✓	✓	✓	✓	x	x	x	x	x	x	x	x	x	x
XFlash–a web application design framework with model-driven methodology	✓	x	✓	x	x	x	x	x	x	x	x	x	x	x	x	x
Leveraging declarative languages in web application development	✓	x	✓	✓	✓	✓	x	x	✓	✓	P	x	x	x	x	x
Grocery CRUD	✓	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
Angular Fullstack	✓	x	x	x	✓	✓	x	x	✓	✓	✓	x	x	x	x	x
Webratio	✓	✓	x	✓	x	x	x	x	x	x	x	x	x	x	✓	x

Table 2-2 Evaluation done for existing prototyping tools

*P indicates that the feature is planned

2.4. Summery

In this chapter, the previous work of the researchers has been critically evaluated by reading their research papers and articles. The advantages of each of the models which were identified has been addressed accordingly and based on that sixteen criteria have been taken into account to identify the research gap. Based on the research gap an in-depth analysis has been carried out as to what criteria has been supported by the previously identified criteria based on the literature review by previous researches. In the next chapter, we will be discussing about the technology which has been adopted in the proposed solution mainly focusing on the deliverables.

Chapter 3

3. Technology foundation of the solution

3.1. Introduction

In the previous chapter an extensive discussion and reviews of researcher's work has been conducted. And we have identified the research gap that needs to be filled. This chapter is a discussion of the technologies which will be used in the proposed solution.

3.2. Technologies used for the solution

Various technologies were used to increase the performance and to ease up the development.

3.2.1. Server-side JavaScript powered by NodeJs

Node.js [10] is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

3.2.2. ExpressJs framework

Express.js [11] is a Node.js framework. Node.js is a platform that allows JavaScript to be used outside the Web Browsers, for creating web and network applications. This means that you can create the server and server-side code for an application like most of the other web languages but using JavaScript.

3.2.3. JSON

JSON [12], or JavaScript Object Notation, is a minimal, readable format for structuring data. It is used primarily to transmit data between a server and web application, as an alternative to XML.

3.2.4. Client-side JavaScript powered by AngularJs

AngularJS [13] is a structural framework for dynamic web apps. It lets the developer to use HTML as the template language and lets the developer extend HTML's syntax to express his application's components clearly and succinctly. AngularJS's data binding and dependency injection eliminate much of the code he would otherwise have to write.

3.2.5. Multiple DBMS support

Sequelize [14] is a promise-based ORM for Node.js and io.js. It supports the dialects PostgreSQL, MySQL, MariaDB, SQLite and MSSQL and features solid transactional support, relations, read replication and more.

3.2.6. Mocha and Chai

Mocha [15] is a JavaScript testing framework, and Chai is a BDD / TDD assertion library. Both Mocha and Chai [16] can run in Node environments or in the browser. In Test-driven development, which means you write your tests before your code, is a great goal to strive for, but takes discipline and planning when you're programming. To make this whole process a lot easier, you need easy-to-use and powerful testing and assertion frameworks, which is exactly what Mocha and Chai are.

3.2.7. Bitbucket

Bitbucket [17] is a web-based hosting service for source code and development projects that use either Mercurial (since launch) or Git (since October 2011) revision control systems that is owned by Atlassian. Bitbucket offers both commercial plans and free accounts.

3.3. Development tools

When implementing the actual solution set of development tools were used to speed up the implementation of the solution.

3.3.1. Visual Code

Visual Studio Code [18] is a source code editor developed by Microsoft for Windows, Linux and macOS. It includes support for debugging, embedded Git control, syntax highlighting, intelligent code completion, snippets, and code refactoring.

3.3.2. PhpMyAdmin

phpMyAdmin [19] is a free and open source tool written in PHP intended to handle the administration of MySQL or MariaDB with the use of a web browser. It can perform various tasks such as creating, modifying or deleting databases, tables, fields or rows; executing SQL statements; or managing users and permissions.

3.3.3. Ampps

AMPPS [20] is a solution stack of Apache, MySQL, MongoDB, PHP, Perl and Python for Windows NT, Linux and macOS. It comes with over 300 PHP web applications, over 1000 PHP classes and various versions of PHP. AMPPS is created by Softaculous Ltd. a company founded in 2009 which makes the Softaculous Auto installer.

3.3.4. mobaXterm

MobaXterm [21] is your ultimate toolbox for remote computing. In a single Windows application, it provides loads of functions that are tailored for programmers, webmasters, IT administrators and pretty much all users who need to handle their remote jobs in a simpler fashion.

3.4. Hosting and Deployment technologies

Since the proposed solution follows service oriented architecture a very sophisticated hosting and deployment technologies were used.

3.4.1. AWS EC2

Amazon Elastic Compute Cloud (Amazon EC2) [22] is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers. Amazon EC2's simple web service interface allows the developer to obtain and configure capacity with minimal friction.

3.4.2. Beanstalk

Beanstalk [23] is the complete code hosting workflow teams or individuals use to write, review and deploy their code. Beanstalk reduces management complexity without restricting choice or control. It can simply upload your application, and Beanstalk automatically handles the details of capacity provisioning, load balancing, scaling, and application health monitoring.

3.5. Summery

This chapter briefed about the technologies and the tools which were used for the proposed solution and the justification of these technologies will be discussed in the later part of this thesis. The next chapter will be discussing on the approach of the proposed solution.

Chapter 4

4. A new approach to fast prototyping

4.1. Introduction

Previous chapter briefed about the technologies which were used in the proposed solution and this chapter will be discussing about the approach to a fast prototyping tool.

4.2. Requirement Gathering

Prior to the initialization of this thesis a considerable amount of time was spent by reading previous research papers and articles done by other researchers. The articles which consisted of great value were taken into account are included in the references. Secondly based on the personal experiences by working in the industry for the past 8 years I personally have faced issues when developing web applications and mobile applications which were considered when developing the solution explained in this thesis. Thirdly ideas were gathered from my working colleagues, project managers, team leaders and industry experts who are engaged in similar work activities not only in Sri Lanka but also in Singapore and Australia.

4.2.1. Issues faced by the project teams working in the industry

When it comes to issues faced by the teams working in the industry we can categorize it into four major sections with respect to software development life cycle. During the planning phase, they spent more time on gathering the correct requirement from the client and multiple client interviews and meetings has to be conducted in order to get the clear picture of the client requirement. A prototyping tool which can generate a prototype based on the user requirement in real time can play handy in this scenario. Because with a working prototype during an ongoing meeting can clear up client requirement more easily on the very first meeting itself. This will reduce the number of meetings which needs to be conducted with the client.

When it comes to the designing phase, system designers and architectures do multiple database designs, UI and architectural designs for the implementation of the best product possible. In this case they come up with several numbers of designs as well as several numbers of fine tuning rounds which consumes more time. As an input from the planning stage a working prototype would make the designing phase easy and faster thus resulting number of design alterations to reduce.

During the implementation phase as discussed in the above chapters developers spend more time on repetitive tasks such as CRUD operations, setting up the based project as well as writing test cases. With a prototyping tool which is capable of generating the customized CRUD operations and write test cases automatically will greatly reduce the time taken on above tasks. Another issue faced by the developers is maintaining the code base which gets more complex when the program is evolving. A good programming framework as well as a good design pattern should be followed to come up with a maintainable code. The software solution which we are implementing is capable of generating a code like that.

4.2.2. Gaps identified which hasn’t been addressed by other researches

An extensive discussion on gaps identified by other researchers in the same domain has been conducted and discussed in the literature review chapter.

4.2.3. Limitations with current development tools

We have discussed on the current development tools also in the literature review chapter and the aim of this research is to address all these major issues faced and identified by us.

4.3. Hypothesis

Hypothesis was built based on the requirements gathered as mentioned above. This research is focused on bridging the gap between the existing issues which are faced by the developers when developing web applications and the current available technologies which were more elaborated in the literature review. Our hypothesis is that service oriented code generator for fast prototyping using schema based requirement definitions.

4.3.1. Fast prototyping

According to the requirements gathered our main goal is to reduce the time taken on development tasks. General Prototyping is only effective during the implementation phase. With the fast prototyping idea, we hypothetically propose a prototyping tool which can be used regardless of the software development life cycle phase. As an example, this tool can be used in a client meeting to generate a prototype in real time. For people who are not familiar with programming technologies can use the cloud based version of the proposed system as a service to generate the code.

4.3.2. Code generators

Proposed system should capable of generating a code based on a user requirement. It should follow a good design pattern such as MVC and generate a source code which follows industry accepted standards. At the same time, it should also generate a source code using a software framework which is more powerful and maintainable. Apart from these features it should also follow a highly model driven architecture which makes it easy to map relationships between data objects.

4.3.3. Prototyping services

Prototyping services usually provide a way of designing the UI and some sort of design diagrams. In our hypothesis, we proposed a prototyping tool as a service which will accept user requirements defined in a file and allow users to download the generated source code of a prototype. Any of the research works that were evaluated did not offer such functionality.

4.3.4. Schema based requirement definitions

Many of the researches conducted, proposed a requirement definition mechanism which is capable of describing the user requirements. Most of them were suggested that defining the content of a UML diagram to a requirement definition file as well as some researchers suggested defining the content of a class diagram to a requirement definition file. All most all of them were using XML as the definition language. Drawback of this approach is that team members have to first design the UML or the class diagram first before building up the requirement definition file. Since our main goal in this research is to reduce the time taken on different tasks, our hypothesis is for a requirement definition schema file which purely focuses on user requirement and not on how the system architecture would look like. This reduces the complexity of the requirement definition file as well as it makes it easier for a novice user to start building up prototypes.

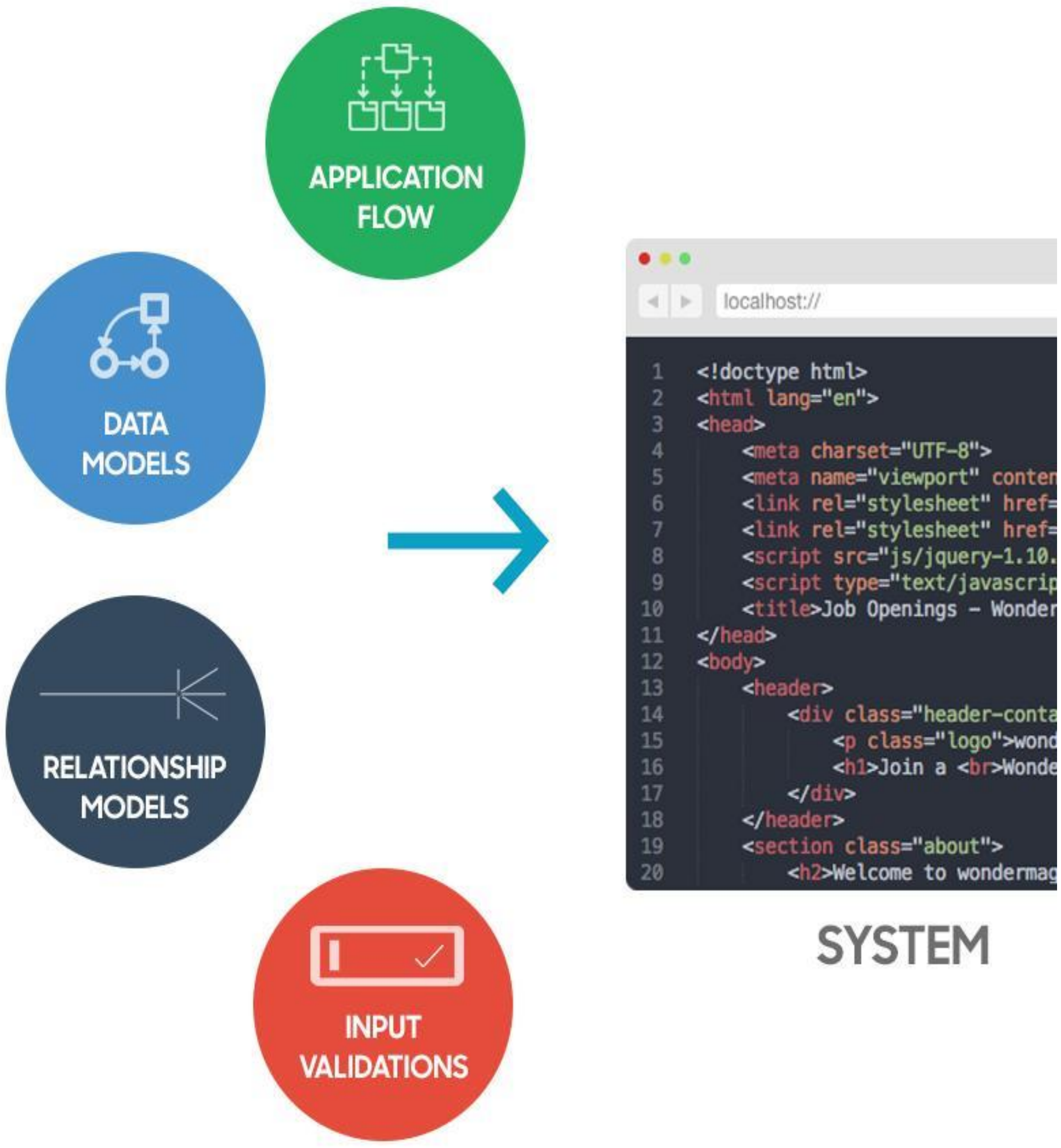
4.4. Users of the system

Target audience of this prototype tool can be ranged from business analysts to software testers.

- Business Analysts – will be able to gather requirements more accurately since they have a product in place when the client meeting is going on.
- Project Managers – will save their time estimating effort, team, cost and time.
- Developers – can use the prototype generated from client meeting straightaway for the development

- Clients (End Users) – are satisfied because they know the development team has got the clear requirement of the system as what they want
- Company owners – By saving developer time owners of the company will make profits than before.

4.5. Inputs to the system



```
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" conten
    <link rel="stylesheet" href=
    <link rel="stylesheet" href=
    <script src="js/jquery-1.10.
    <script type="text/javascrip
    <title>Job Openings - Wonder
</head>
<body>
    <header>
        <div class="header-conta
            <p class="logo">wond
            <h1>Join a <br>Wonde
        </div>
    </header>
    <section class="about">
        <h2>Welcome to wondermag
```

Figure 4-1Inputs to the proposed system

As we discussed before most of the researches suggested that a customized requirement definition is helpful in generating the most optimal and customized code for a given requirement. keeping that in mind and based on inputs from the industry experts as well as based on personal experiences a schema definition model was build. goal of implementing our own customized requirement definition schema was simplify the requirement definition process unlike accepting class diagrams or UML diagrams. making the requirement definition more complex means that it is much harder to use by a novice user. primary goal of this research is to reduce the time taken on each development task.

By collecting data from others, we have noticed that set of requirement areas needs to be fulfilled in order to generate a complete prototype with maximum possible requirements covered. according to diagram shown above requirement definition should cover the application flow, data models, relationships between data models as well as the user input validations. This model leads us to two main definition schema files. namely flow definition schema and data model definition schema.

4.5.1. Flow Definition Schema

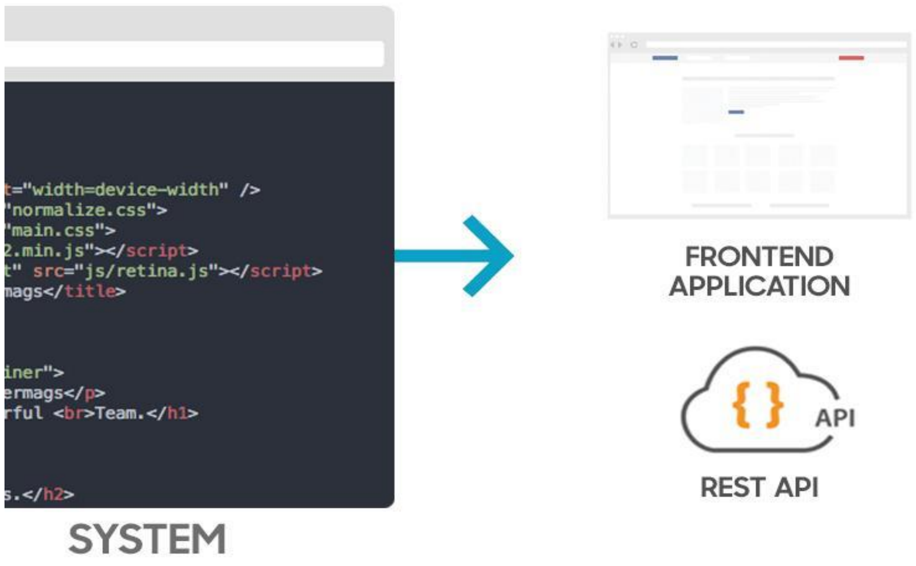
One main feature identified from the literature review which needed to be implemented was that introducing a customized system flow definition mechanism. in this scenario, we had to research on previous works as well. without inventing our own thing, we thought of getting the base idea from angular UI router [24] which is an open source project. it was modified according to our need lately.

4.5.2. Data Model Definition Schema

Another requirement was to get the user requirement for data models in which by using we can generate the database of the application. apart from getting only data model data, we found that data model definition schema can be used define customized user input validations as well as the relationships between each data models.

4.6. Outputs of the system

Output of this prototype generator is a client-side application using single page web application technology and a backend REST API [25] which provides data to the client side application to function. database is built automatically when the REST API starts running as service in server side as it has all the data from the data model definition to build the complete database. separation between client side application and server side allows us to support mobile application technologies as well.



```
t="width=device-width" />
"normalize.css">
"main.css">
2.min.js"></script>
t" src="js/retina.js"></script>
mags</title>


iner">
ermags</p>
rful <br>Team.</h1>


s.</h2>
```

Figure 4-2Outputs of the proposed system

4.7. Features

When it comes to features of proposed prototype generator, we can discuss the features of it in two main categories,

User Specific Features

Novel users can also use the standardized JSON schema for building a prototype application. simplified requirement definition makes it much more productive than conventional systems used in the industry. Since all the ground up work is already done beforehand, project start up time is reduced considerably. Source code which is generated using proposed system meets all the industry standards. And minimal of errors since the code which it generates is already tested.

System Specific Features

This prototyping tool planned to support code generation in multiple languages such as PHP, NodeJs, C#, JSP and ASP.net. Since the generated application uses popular web technologies, it also supports multiple platforms such as Linux, Windows, Mac.

Multiple DBMS system support

Proposed generated uses popular and intelligent ORM system from the industry which enables it to support multiple databases such as SQL, MySQL, Postgres and NoSQL databases such as MongoDB.

additionally, features like secure Authentication and Authorization, Social Media Integration is in just few customizations away.

4.8. Summery

This chapter explained the approach which allowed us to come up with a solution which is able to fill the research gap. This enabled up to come up with a hypothesis that, by using two different client requirement definitions schemas, we can generate the source code for a fully featured prototype application. And we have discussed the relevant steps/ approaches as well as the components required for the solution. In the next chapter will be discussing on the system design in detail.

Chapter 5

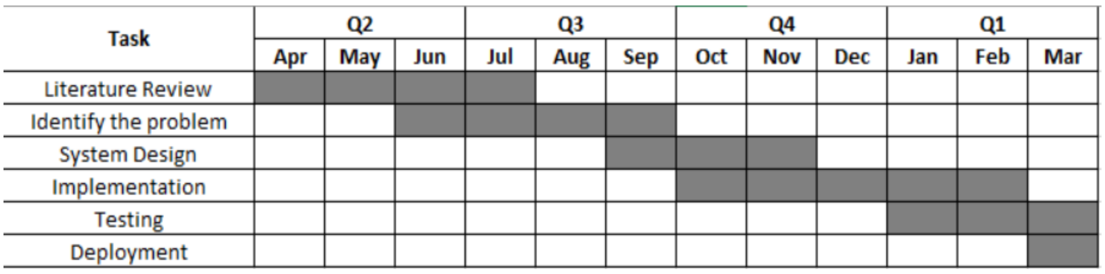
5. Analysis and Design of the new prototyping solution

5.1. Introduction

The previous chapter gave a full picture of the approach to a new prototyping tool. This chapter describes the design of the solution for the process presented in the approach. We design the solution which will generate a prototype application where it will work as a client-server system with a backend database. Here we describe the top-level architecture of the design by elaborating on the role of each component of the architecture. We will discuss about the System design, platform design, Infrastructure design of the system throughout this design chapter.

5.2. Research planning

Planning and the scheduling of the project is shown in the table below. Most of the time were spent on literature survey and the implementation stages



Task	Q2			Q3			Q4			Q1		
	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar
Literature Review	■	■	■	■								
Identify the problem			■	■	■	■						
System Design						■	■	■				
Implementation							■	■	■	■	■	
Testing										■	■	■
Deployment												■

Figure 5-1 Execution plan for the proposed system

5.2.1. Development methodology

Evolutionary prototyping methodology was used for implementing the system, because of the research component involved in the project. Number of fine tuning rounds were required to get the best possible prototype generated.

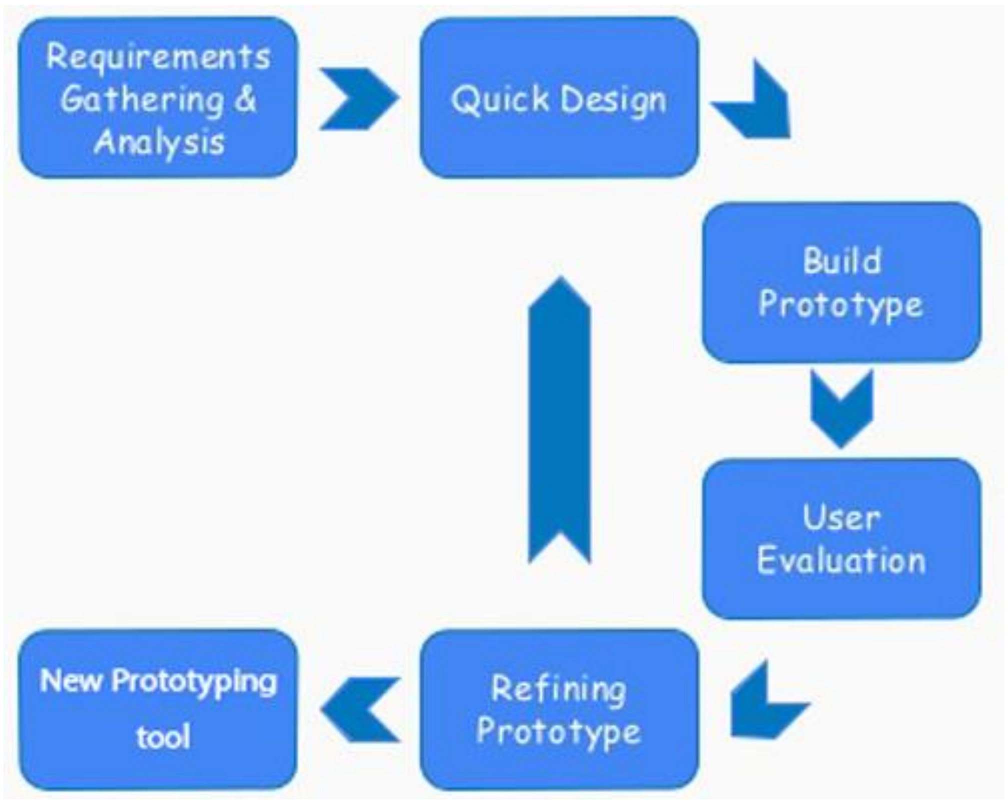


Figure 5-2Evolutionary prototyping methodology used for proposed system

5.2.2. Selection of the software process model

Before selecting the process model for this research, a considerable time was spent on literature review. By that we realized that the suggestions which were given by previous work was not the same most of the time. Because of that we had to test most of the suggestions and work on the best possible method.

5.3. Analysis of the current development workflow

As discussed in the approach chapter, prior to the initialization of this research good amount of time spent on examining the previous research papers. Based on the personal experiences by working in the industry for the past 8 years I personally have faced issues when developing web applications and mobile applications which were considered when developing the solution explained in this thesis. Ideas and suggestions were gathered from my working colleagues, project managers, team leaders and industry experts who are engaged in similar work activities.

5.4. Requirement analysis

Application usage can be described in few simple steps.

- User defines the flow and the data models using requirement definition schema. Then user uploads the file to system (cloud based service) or user can specify the schema files (for real-time version)
- System processes the input files and its requirements.
- System allows user to down load the generated source code of a prototype.

5.4.1. Functional requirements of the solution

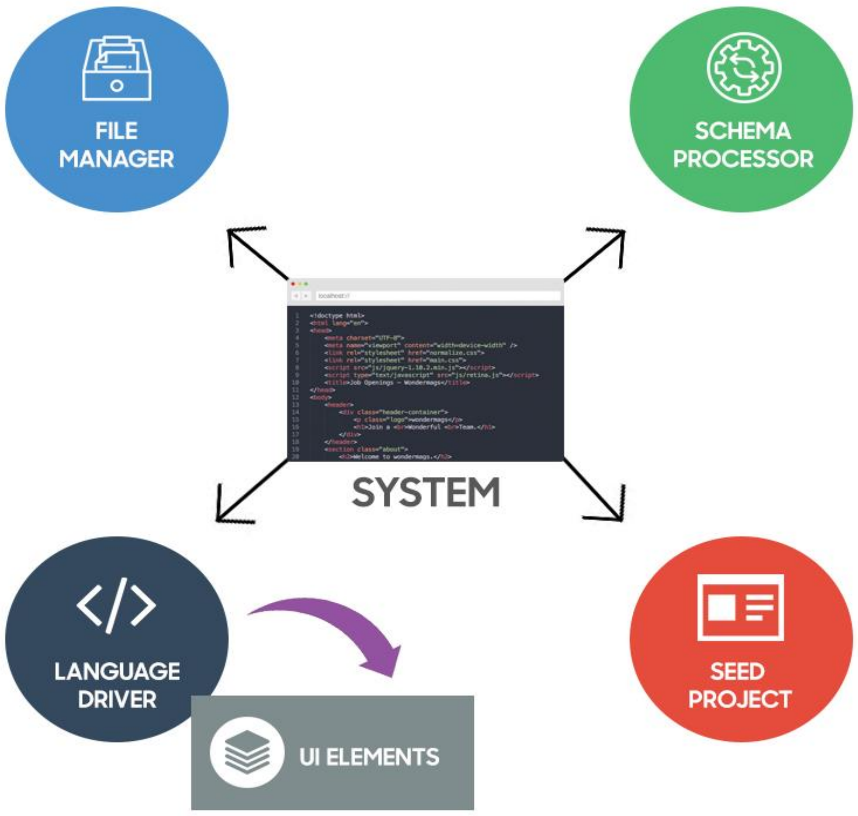
Main goal of this research is to implement a code generation for fast prototyping tool, it should allow users to generate customized application flow. Following the Model-Driven architecture is a must as it allows relationship mapping and ORM usage. This allows the system to make it supportable for customized data model generation. Generated code needs to be in MVC design pattern. Following MVC architecture makes it easier to generate a app in a modularized fashion. Generated application should be using industry accepted and widely used frameworks. Backend should be consisting of a REST API as well.

5.4.2. Non-functional requirements of the solution

Real-time code generation is a plus since the goal of the project is to reduce the time spend on coding. Generated software should be optimized for performance as well as it should be in the production grade quality. Maintainable code is a must in this case as the prototype which gets generated can be used as the base code for a project.

5.5. Top level design architecture

As the figure shown in below, top level design architecture of the system consists of 4 main modules. Which is the file manager module, schema processing module, language driver module and the seed project which contains all the necessary seed files for generating the prototype.



```
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width" />
    <link rel="stylesheet" href="normalize.css">
    <link rel="stylesheet" href="main.css">
    <script src="js/jquery-1.10.2.min.js"></script>
    <script type="text/javascript" src="js/retina.js"></script>
    <title>Job Openings - Wondermags</title>
</head>
<body>
    <header>
        <div class="header-container">
            <p class="logo">wondermags</p>
            <h1>Join a <br>Wonderful <br>Team.</h1>
        </div>
    </header>
    <section class="about">
        <h2>Welcome to wondermags.</h2>
```

Figure 5-3 Top level design architecture of the proposed system

5.6. Module architecture

As we discussed the proposed prototype generator consisting of 4 main modules. We will further elaborate on each of these modules

File Manager

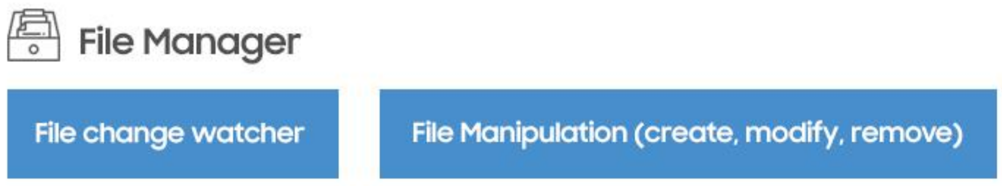


Figure 5-4Modules contained in the file manager

File manager consists of 2 major functionalities. System uses file manager module to watch for file changes and restart prototype building process automatically. This capability of file manager allows us to introduce real-time prototype generation. Primary functionality of the file manager is to manipulate files. Namely creating new files, writing generated code for files, removing temporary files, maintaining the folder structure, maintain the distribution flow and many other things to file processing.

Schema Processor

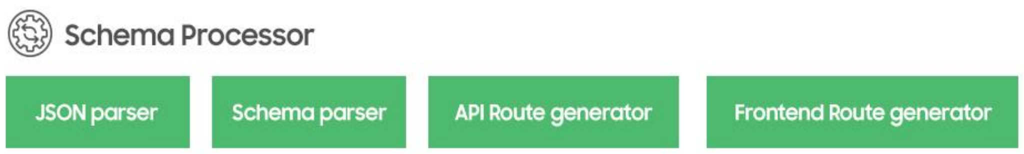


Figure 5-5Modules contained in the schema processor

Schema processor is the heart of the proposed system. This consists of 4 main functionalities. JSON parser, Schema parser, API route generator and the Frontend

Route generator. JSON parser is responsible for taking file content from the file manager as input and process the JSON content inside the file. This involves converting user defined requirement to specific flow data as well as processing data which can be used to implement the database on top of an ORM. Processed data is then passed in to different sections of the system which then decides on REST API URLs as well as the client-side browser navigation (frontend route generator).

Language Driver

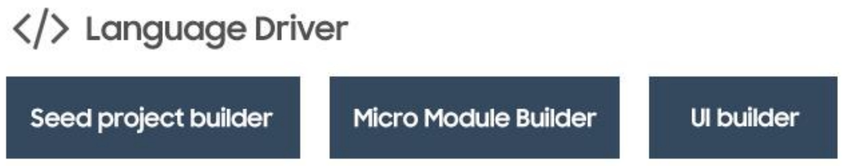


Figure 5-6Modules contained in the language driver

Once the processing the schema which user has provided, its then transferred to the respective language driver. Proposed system is capable of supporting multiple languages. Client-side application is powered by angularJs. initially backend REST API is generated using NodeJS using expressJs framework. Each of these languages have a different language driver. When it comes to the characteristics of this language driver each of these drivers contains a seed project. This seed project was taken from the well-defined and stable product source codes. This makes it bullet proof when it comes to stability maintainability as well as the quality of the generated output. Language driver modifies the seed project according to the inputs from the schema processor. It also builds the requirement specific modules and places inside the seed project. Process is valid for both client-side and the server-side applications. Language drive is also responsible for generating the customized UI elements with the user input validation support.

Seed Project

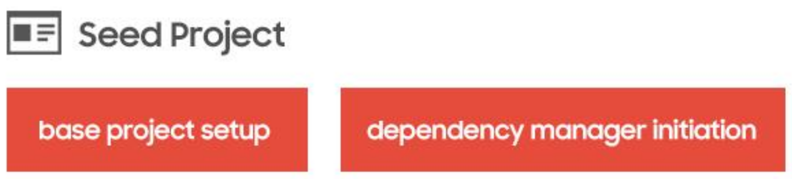


Figure 5-7Modules contained in the seed project

As discussed above, seed project is taken from one of the most stable product sources codes which is available in the market. These seed projects are highly model driver while following the MVC design pattern. This makes the generated prototype a MVC model driven application. System also divides the user requirement in to modules. And it makes it easier to maintain and understand the code easily. This also provides the support for modular code base architecture. Seed project is also powered by

5.7. Schema based requirement definition

As we discussed before our solution required to find a user requirement definition this lead us to research on a better requirement definition mechanism. Most of the researchers which has been conducted was using XML as the requirement definition language. this makes it much more complicated for a novice user to understand and learn, because of this complexity. According to the suggestions, which we have got from the industry experts, we have decided to use JSON as the requirement definition language because of the routing module we are using which is angularUI router.

5.7.1. Application flow definition

Application flow definition is done using JSON. It can define its different states and by using this angualarJS UI router. By using this module, we can define the state parameters then and there. The schema file uses natural language which makes it easier to understand by a non-technical user. each and every state has its own unique ID. user can define its title as well. if user requires a customized template user defined as well. another feature which includes in this is that use of child states. Each parent state can have any number of child states. this makes it the prototype highly customizable. apart from that users can define on how the data should be displayed in the client side. related data models can also be defined using flow definition mechanism. We will be discussing further more on how the application flow definition is implemented in the implementation chapter.

5.7.2. Data Model Definition

Proposed system requires a data model definition which can be used to generate the database. according to the research which we have conducted, we identified that not only data models but also user input validations can also be defined. as well as appropriate messages and the data types of each of these models can be defined using data model. another advantage of this data model over previous researches conducted is that we can also specify the UI element which needs be used on each of the data field by using this data model. you scan specify the database connection string which will be used in the generated prototype. Additionally, users can define the relationships between different data models. at the initial phase, this supports has-one, has-many and belongs-to relationships.

5.8. Generated Prototype

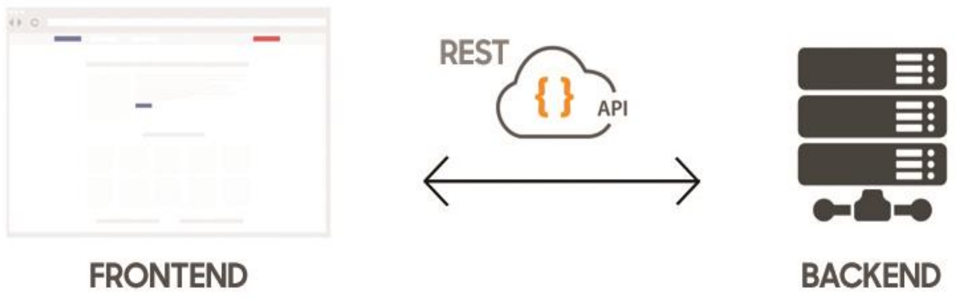


Figure 5-8 Generated prototype top level design

As we discussed previously generated prototype consists of two major sections. first one is the client-side application the second one is rest API which provides data to function the client-side application. both of these applications use latest technologies which allows users to focus more on implementation rather than focusing on different Technologies. The generated rest API supports mobile authentication as well. this enable developers a hassle free mobile application implementation. as we discussed before generated frontend is also powered by Angular JS which is the most popular single page application Framework for JavaScript nowadays. If users require the front end in a different Framework it's just a language driver away. initially backend is powered by node JS. The generated prototype API is built on top of Express JS framework and also it uses sequelize as the ORM. because of its clever query builder, we don't have to worry about on how the data is read and retrieved from the database

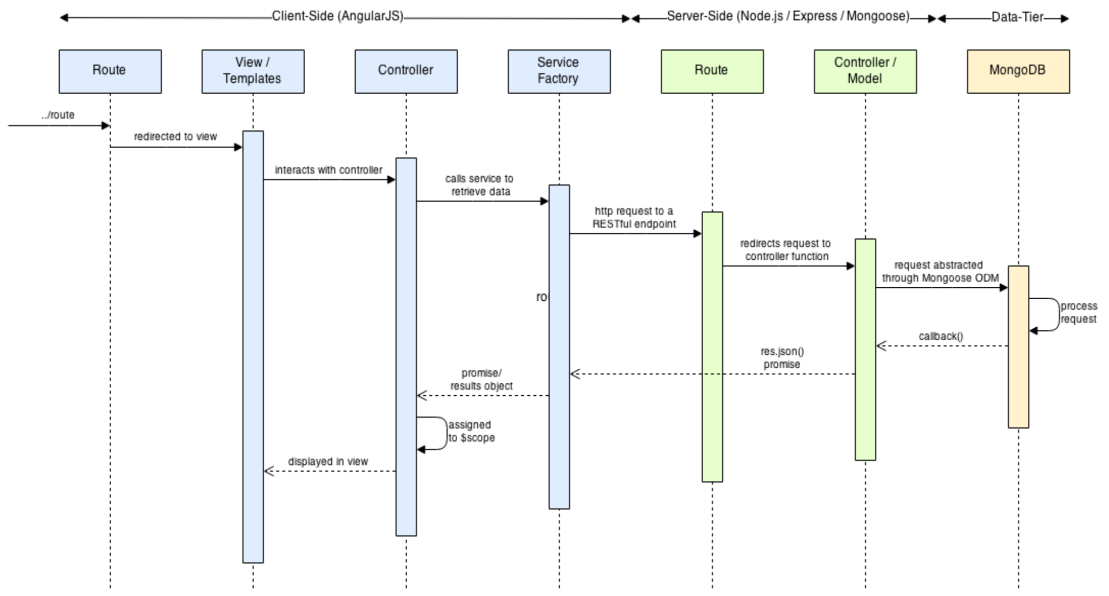


Figure 5-9 Detailed view of the data communication between generated client-side and the server-side application

5.8.1. Generated client-side application design architecture

Following diagram describes on how the client side angularJs application works with its dependencies as well as the modularized code.

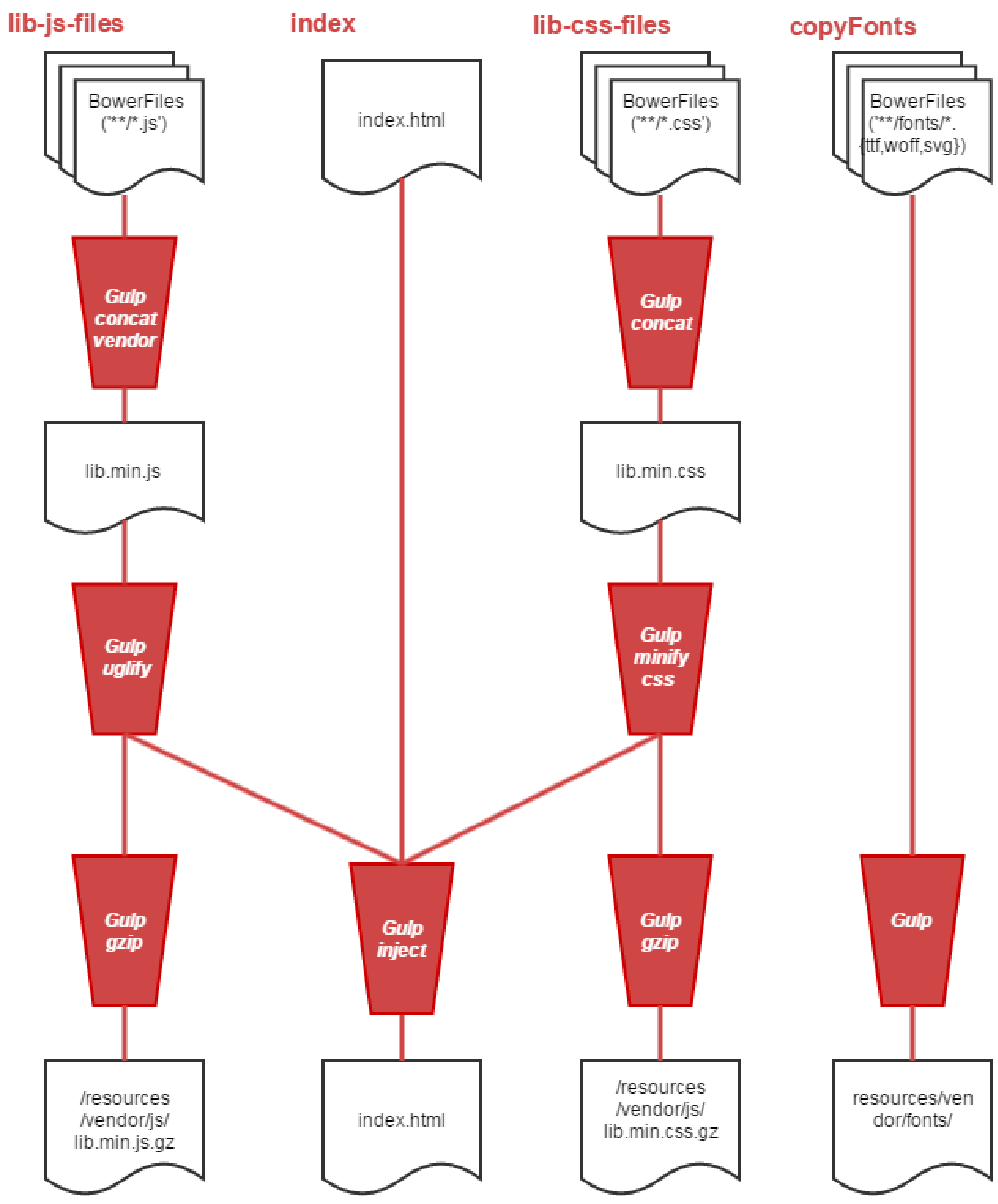


Figure 5-10Process of the client-side application working with gulp task runner

5.9. Dependency management, ground up work on generated prototype

A good prototype generator should be able to the most of the ground of work on behalf of the developer. primary goal of a prototype application is to reduce time taken on different tasks. when we discuss over good prototyping tool, we can characterize it over set of highlighted functionalities which it should provide,

5.9.1. Version controlling

Proposed solution provides out of the box version controlling system which supports integration with popular services such as bitbucket or GitHub. It is always recommended to use version controlling on a project before even starting up the project. This is one of the features which was not included in most of the researches which we have read on.

5.9.2. Dependency management

For this product, we have used npm as our dependency manager. because this solution is built using node JS. apart from that the prototypes which gets generated are also using its own dependency managers. client-side application uses bower as its dependency manager. this is the most popular dependency manager for client-side web applications with Angular JS version 1. generated rest API is built using node JS and it uses npm as the dependency manager. usually setting up these dependency managers, installing as well as deciding on the required dependencies takes around 5 to 12 hours depending on the project complexity. We have bundled up all the required dependencies inside these generated prototypes. this makes sure developers can focus on implementing the solution rather than spending extra time on initial set up time.

5.9.3. Compiling and building

Even though we use JavaScript with node JS, JavaScript is outdated now and the new version of the JavaScript is the ES 6. To support ES6, we use babel as the superscript of JavaScript which later automatically gets compiled into JavaScript. For the client side application, we use typescript which is also a superscript JavaScript and industry accepted coding standard for front end web applications. Using typescript, we get an easily maintainable code.

5.9.4. Testing and continuous integration

For making the building and continuous integration process faster we use gulp task runner [26] in the generated prototype. And as the testing framework on generated prototype, it gets generated using mocha and chai testing frameworks.

5.10. Summery

In this chapter, a detailed description the system design has been discussed. Usage of different technologies was also mentioned in this chapter. In the next chapter, we will be discussing about the implementation of the proposed system according to the designs which we have discussed

Chapter 6

6. Implementation

6.1. Introduction

In the previous chapter, we discussed about the full picture of the entire solution and we discussed system architecture of the proposed solution. In this chapter, we look in to our solution implementation. Proposed solution consists of four major modules and we will be discussing on how these modules are implemented as well as this chapter will be discussing on the implementation of the generated prototype.

6.2. Overall solution

Overall solution has been implemented as an open source application that can be accessed by any client running on any OS including Windows, Linux or MacOS. The built prototype applications are based on client-server architecture. And it supports multi language as well as multi-platform code source generation.

6.3. Implementation of the Solution

Proposed solution is a collection of software modules, in which each module consisting different sub modules. These modules are using different technologies, mentioned in the technology chapter (chapter 3) to achieve its goal and pass the results to the next module. Final outcome of the solution is a prototype web application source code which is generated using user defined output language (PHP, NodeJS, etc.).

6.3.1. Preparation

Before starting of the implementation all the ground up work was done as we discussed in the previous chapters. Such as setting up version controlling setting up build processes, setting up testing mechanism, etc. Finding good seed projects for language drivers was also a pre-preparation task as the generated prototype needed number of fine tuning rounds. Study on REST APIs was also conducted before beginning the implementation of the proposed solution. This helped us on designing fully featured and secure seed project for server-side application.

6.3.2. Programming

For programming, we have used the Visual Code IDE which provides set of comprehensive tools to work with in open source projects. And it runs on any platform. System was implemented using node js. Nodemon plugin was to automatically restart the servers on code changes. For easy data manipulation on database we have used PHPMyAdmin which provides easy access to the database. For running mysql server we have used ampps server.

6.4. Requirement definition processing stored in schema files

As we discussed during design chapter, two requirement definition schema files were proposed. According to the design of the system we have implemented the schema structure. One for defining the flow of the system. And another one for defining the data models of the system. Both are described in detail below,

6.4.1. System flow definition mechanism

As shown in the following code snippet, it shows a sample application flow definition for a small library management system. Using this file user can define its different states also we can define the state parameters. The schema file uses natural language which makes it easier to understand by a non-technical user. each and every state has its own unique ID. user can define its title as well. if user requires a customized template user defined as well. another feature which includes in this is that use of child states. Each parent state can have any number of child states. this makes it the prototype highly customizable. apart from that users can define on how the data should be displayed in the client side. related data models can also be defined using flow definition mechanism.

```
{
    "id": "main",
    "title": "Welcome to FES (Forward Engineering System)",
    "template": "welcome.html",
    "linksTo": [
        "main.home"
    ],
    "children": [
        {
            "id": "home",
            "title": "Library System Home",
            "template": "home.html",
            "linksTo": [
                "main.authors",
                "main.books",
                "main.users"
            ]
        },
        {
            "id": "authors",
            "title": "Book Authors",
            "linksTo": [
                "main.home"
            ],
            "sections": [
                {
                    "title": "Users",
                    "useModel": "user",
                    "for": "list",
                    "display": "list"
                }
            ]
        },
        {
            "id": "books",
            "title": "Library Books",
            "linksTo": [
                "main.home"
            ]
        },
        {
            "id": "users",
            "title": "System Users",
            "linksTo": [
                "main.home",
```

Figure 6-1Sample Application Flow Definition Schema

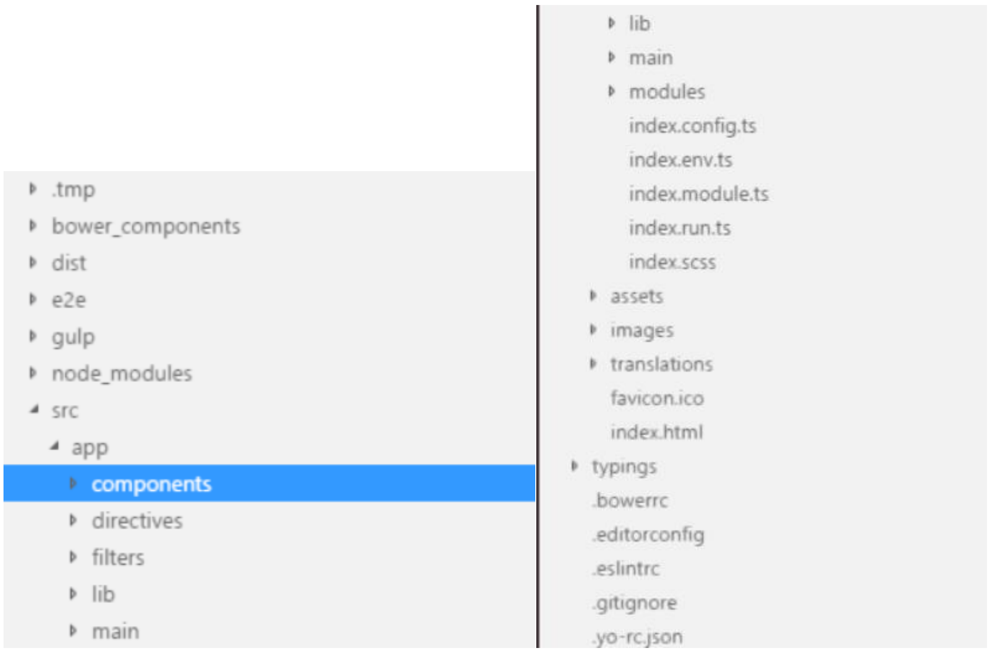
6.4.2. Data Model Definition

```
{
    "name" : "FES DB Schema",
    "connectionString": "mysql://root:mysql@localhost/fes",
    "models": [
        {
            "name": "user",
            "fields": [
                {"name": "firstname", "type": "String", "element":"text", "vali-
dation": {"required":"Firstname is required"} },
                {"name": "lastname", "type": "String" , "element":"text"},
                {"name": "address", "type": "Text" , "element":"textarea"},
                {"name": "email", "type": "Text" , "element":"textarea", "valida-
tion": {"email":"Valid email should be entered", "required":"email is re-
quired"}},
                {"name": "age", "type": "Number" , "element":"text"},
                {"name": "gender", "type": ["Male","Female"] , "element":"check-
box"},
                {"name": "city", "type": [ "Colombo", "Moratuwa", "Panadura",
"Mount Lavinia", "Kandy" ] , "element":"select" }
            ]
        },
        {
            "name": "permission",
            "fields": [
                {"name": "name", "type": "String" },
                {"name": "value", "type": "String" }
            ],
            "relationships": [
                {
                    "type": "hasOne",
                    "model": "user",
                    "as": "user",
                    "reversAs": "permission"
                }
            ]
        },
        {
            "name": "author",
            "fields": [
                {"name": "firstname", "type": "String" },
                {"name": "lastname", "type": "String" },
                {"name": "country", "type": "String" },
                {"name": "male", "type": "Boolean" }
            ]
        },
```

Figure 6-2Sample Application Data Model Definition Schema

As shown in the above figure we have implemented a model definition mechanism which is capable of defining the data models related as well as each field of it, data type, UI element which needs to be used in UI generation, input validation as well as the proper messages for input validations. according to the sample code provided relationship between data models can also be defined.

6.4.3. Seed projects



```
▸ .tmp
▸ bower_components
▸ dist
▸ e2e
▸ gulp
▸ node_modules
◢ src
  ◢ app
    ▸ components
    ▸ directives
    ▸ filters
    ▸ lib
    ▸ main
```

```
    ▸ lib
    ▸ main
    ▸ modules
      index.config.ts
      index.env.ts
      index.module.ts
      index.run.ts
      index.scss
  ▸ assets
  ▸ images
  ▸ translations
    favicon.ico
    index.html
▸ typings
  .bowerrc
  .editorconfig
  .eslintrc
  .gitignore
  .yo-rc.json
```

Figure 6-3Generated prototype structure

Shown above is the generated application structure. You can see that generated prototype for frontend consist of source folder dependency management folder which are node_modules and bower_components. inside the source folder we have the modules folder which need to be modified by the developer for customized business logic. language driver copies generated modules inside the model's folder. additionally, generated prototype Support Technologies such as SASS, webpack, HTML5 and translations as well.

6.5. Language drivers for source code generators

Language driver plays a major role in generating the source code for the prototype.

6.5.1. Multiple server-side languages support

Our implementation support for multiple languages though initially it supports for NodeJS only. Following figure is the structure of the language driver.

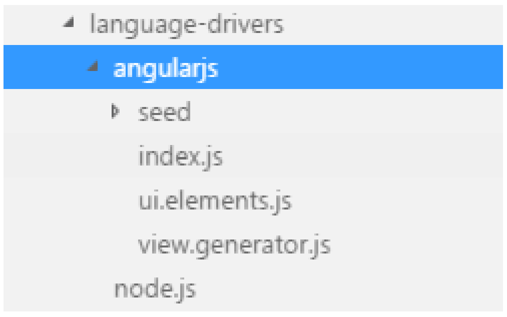
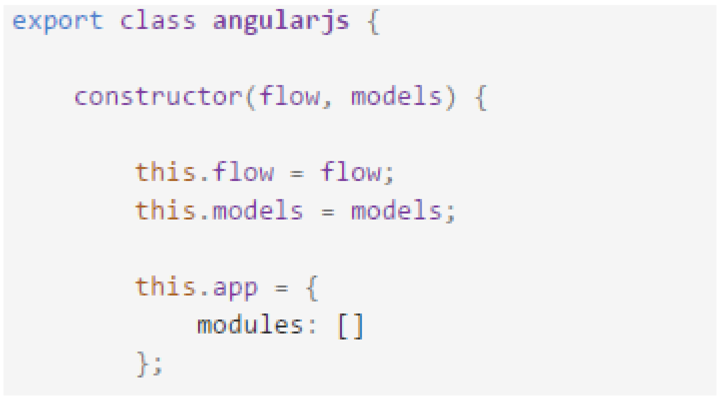


Figure 6-4 Language Driver Structure

As you can see language driver consists of seed project, UI element generator as well as the module builder.



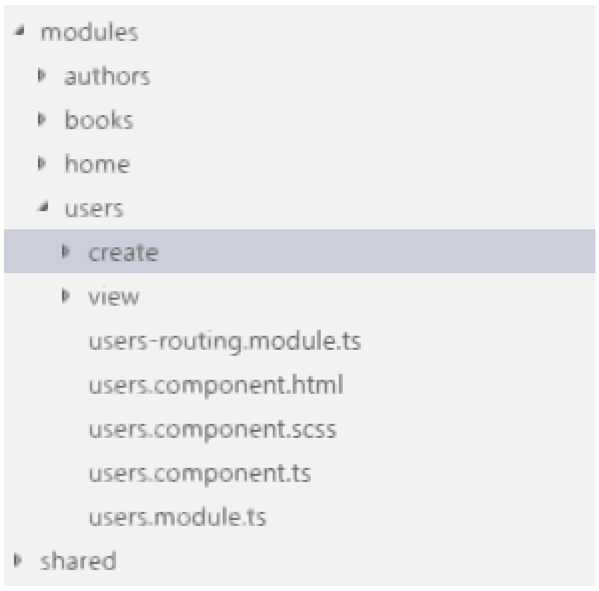
```
export class angularjs {

    constructor(flow, models) {

        this.flow = flow;
        this.models = models;

        this.app = {
            modules: []
        };
```

Figure 6-5Language Driver Constructor

Constructor of each of these language drivers takes the processed schema files as inputs and uses them for code generation.

6.5.2. Modularized architecture

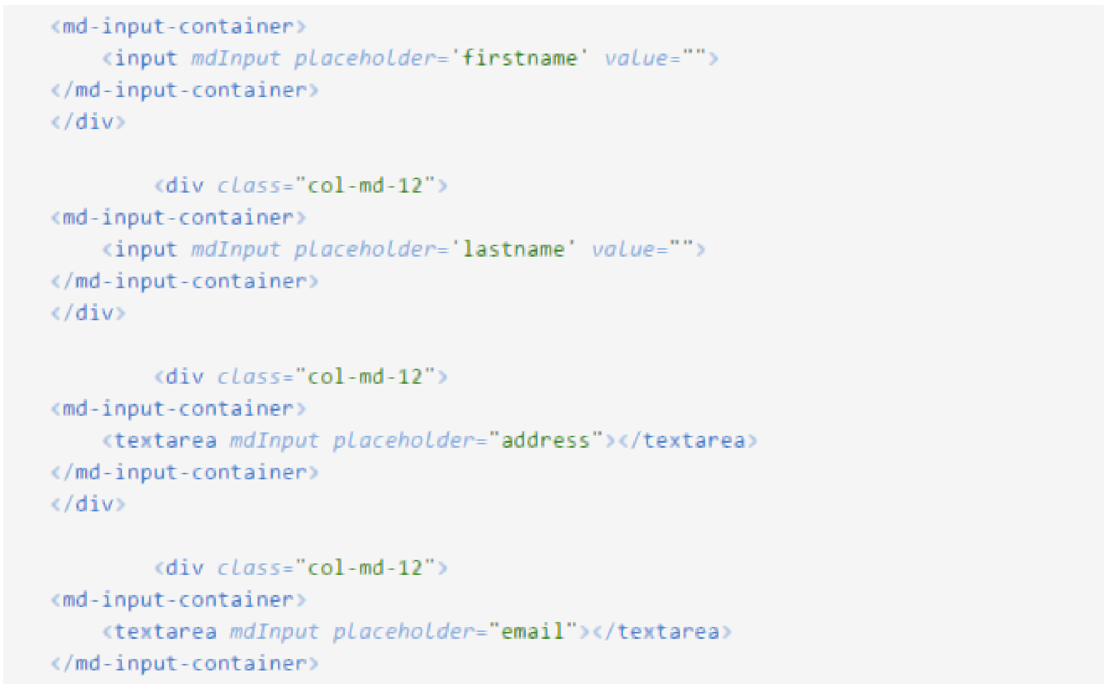
Generated prototype follows the modularized architecture.



```
◢ modules
  ▸ authors
  ▸ books
  ▸ home
  ◢ users
    ▸ create
    ▸ view
      users-routing.module.ts
      users.component.html
      users.component.scss
      users.component.ts
      users.module.ts
▸ shared
```

Figure 6-6Example of generated modularized architecture

As you can see for each of the state which has been defined in the flow definition file, it has created a separate folder as a module. Each of these modules contains the routing data UI elements as well as styling information with it.



```
<md-input-container>
    <input mdInput placeholder='firstname' value="">
</md-input-container>
</div>

        <div class="col-md-12">
<md-input-container>
    <input mdInput placeholder='lastname' value="">
</md-input-container>
</div>

        <div class="col-md-12">
<md-input-container>
    <textarea mdInput placeholder="address"></textarea>
</md-input-container>
</div>

        <div class="col-md-12">
<md-input-container>
    <textarea mdInput placeholder="email"></textarea>
</md-input-container>
```

Figure 6-7Generated UI source code based on data model

This figure shows how the UI input elements are generated based on the definition of the data model schema.

6.6. Generated sample user interfaces

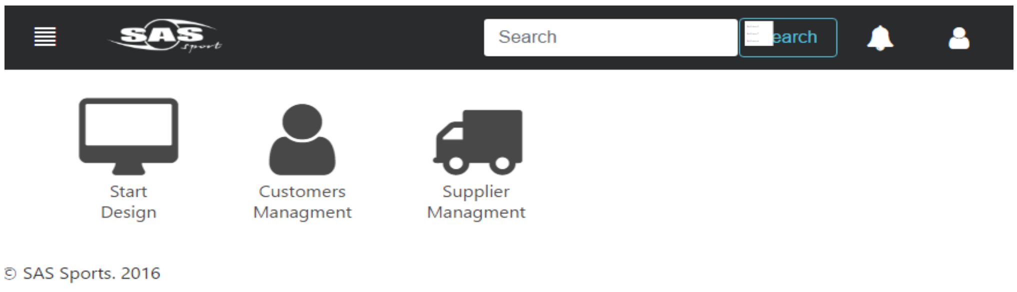


Figure 6-8Sample generated UI

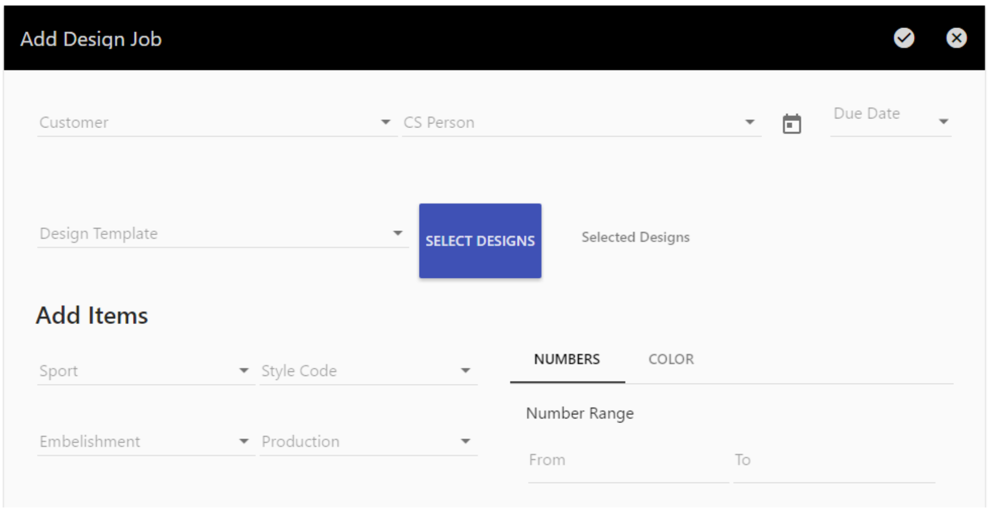


Figure 6-9 Sample generated UI

6.7. Summery

This chapter was about implementation of the proposed solution. It described about the implementation of the major modules that it has and about the functionalities which each module is equipped with. Next chapter will describe on how the evaluation was done of implemented system and the details of whether the objectives have been achieved using test cases. Further it'll discuss on drawbacks and limitations of the implemented system.

Chapter 7

7. Evaluation of the solution

7.1. Introduction

In this chapter, we will be discussing about how the implemented solution was evaluated using two case studies namely case study 1 feasibility for a library management system and case study 2 – service oriented prototyping support stated below. These two case studies were developed in order to evaluate the implemented solution. Apart from evaluation based on case studies, another observation based evaluation was also conducted among set of developers. This chapter will also discuss on how the data was gathered and how it was analyzed using the observation method and has been compared with previous daily work carried out by the developers and their experience after implementing with our system.

7.2. Case study 1 – Feasibility for a library management system

7.2.1. Problem definition

The first case study which we have conducted is a generation of a simple library management system. This library management system is consisting of users, books, authors as well as staff. Users should be able to borrow 1 or more books. Each book has an author. And staff can manage users, books. Staff plan to implement a mobile application in the future. For the same purpose.

7.2.2. Requirement analysis

Above mentioned system and its requirement should be full filled. Proposed system should contain CRUD functionality for users, books and its authors. Relationships should properly be mapped and data integrity should be maintained. According to the scenario it is a plus point if the generated prototype can be used with mobile application development as well. To support this, it should generate a REST API.

7.2.3. Design and Implementation

Building up the prototype process was observed under three circumstances. which are, hand coding the library management system. Using a crud automation system and thirdly using our fast prototyping tool. Observation on each implementation approach was conducted. And results were collected based on predefined evaluation criteria.

7.2.4. Evaluation

Method

To complete the evaluation, set of evaluation criteria were decided which was used to evaluate the solution. After that a critical evaluation based on these evaluation points were being conducted.

Results

Evaluation	**Manual Coding**	**Popular CRUD automation systems**	**Our fast prototyping tool**
Customized flow generation	Customized flow generation is not possible. But any flow can be hand coded. Time taken on this task is pretty high	Customized flow generation is not possible. Only CRUD operation interfaces are generated	Yes. Customized flow can be generated.
Model-Driven architecture	Yes. Model driven approach is possible. Again, the time taken on programing to support model-driven architecture is high.	Yes. Was very fast in generating the code with necessary data models	Yes. Was very fast in generating the code with necessary data models
MVC design pattern	Yes. time taken on programing is high.	For Some CRUD generators, Yes. Was very fast in generating the code	Yes. Was very fast in generating the code
Relationship mapping and ORM support	Yes. time taken on programing is high.	For Some CRUD generators, Yes. Was very fast in generating the code	Yes. Was very fast in generating the code
Server-side REST api support	Yes. Research and following best practices takes a very long time. And experienced developers	No	Yes. Was very fast in generating the code
Industry accepted framework usage	Yes. Again, developer needs to research on coding and best practices as well as setting up the initial development work	No. except very few researches	Yes. Was very fast in generating the code

Time spent on the project	3 days. This included initial ground up work. Setting up and installing dependencies. And various other initial state tasks which we have discussed in this thesis	Generation of crud completed in about 2 hours. But again, time had to spend on setting up the project as well as customizing the generated code to support related data manipulation. This extended the time taken by using a CRUD automation time to 1.5days	Using the prototyping tool which we have developed. Complete system was able generate in 20 minutes. Customization to the UI as well as fine tuning the generated code took about another 1 hour. Which makes it a 2 hours maximum to generate the required prototype.

Table 7-1Evaluation conducted in the case study 1

As expected manual code took the longest time which is about to complete the project. Because they worked in a traditional way and was not using the modern technology. CRUD automation systems were quick in automating the CRUD automations but they were lagging behind when it comes to

7.3. Case study 2 – Service Oriented Prototyping and Code generation

Case study 2 was conducted based on service oriented prototyping capability of the tool which we have proposed. Main goal of this research is to reduce the time taken of different tasks. Apart from that it should be able to use by a novice user. And simplicity is the key.

7.3.1. Problem definition

Usually a client meeting is conducted by a business analyst or a project manager where they get the requirement from the client and pass it on to tech leads and developers. To clear up the requirement multiple client meetings are conducted. Some times when a developer is available they can take the developer with them. Where the developer also gets requirement and check for the technical feasibility of the client requirement. Goal is to achieve this without spending the time of a developer on a client meeting. In this purpose, there should be a prototyping mechanism where anyone can generate a prototype using a simple tool and show it to the client and get the confirmation then and there. Even without setting up a prototyping tool in the local machine itself. Cloud based solution is preferred.

7.3.2. Requirement specification

According to the scenario, there should be a prototyping tool which runs on the cloud. Which should allow people to prototype and download the generated source code which can then be run on a local machine.

7.3.3. Design and implementation

Same as the previous case study, an evaluation was done by comparing two available products. We tried to create prototypes using available cloud based solutions and tested the capabilities. Generation of above mentioned library management system is the goal.

7.3.4. Evaluation

Method

Evaluation was done by evaluating the 2 available products under different criteria's which are mentioned below.

- Invision [27]
- ProtoIo [28]
- MockUps [29]

Results

Unfortunately, none of the research papers which we have evaluated in the literature review does not support cloud based prototyping. In this case we will be evaluating our tool with other prototyping tools which are available in the market with cloud support.

Evaluation	Other available tools	Our fast prototyping tool
Support for automatic code generation	No. any of the available online prototyping tool does not provide a functionality to generate automatic code generation	Yes
Support for customized flow definition	Yes. Even though they don't have automated code generation tools which are available, capable of handling customized flow generation. This is only for client-side	Yes. For both client-side and server-side
Automating CRUD operations	No	Yes
Interactive designer	Yes. All of these tools interactive designer which mostly focuses on customized UI	No. our proposed system doesn't include an interactive designer at the moment
All the evaluations from the case study 1	No	Yes
Time taken on generating a prototype	Users were able to generate a prototype in about 3 hours and they mostly spend time on designing the UI.	People were able to generate and download the application in about 20 mins. Though they needed to install dependencies separately once the project is downloaded. UI was not customizable as much the other tools which we used.

Table 7-2 Evaluation Conducted in the case study 2

We identified that a prototyping tool which works on cloud is a requirement which needs to be fulfilled. None of them were able to generate a REST API or CRUD operations which is a must for developers who are working in the industry. Research improvements also identified such as interactive designer.

7.4. Other evaluations

Here we will discuss on evaluating the implemented solution in other ways. And the evaluation method is observing the usage of real world developers using the system to generate the first level (application prototypes) of the code of their client projects. This will go through the test cases which were used to evaluate the system. Further we will discuss whether the system meets the goal and objectives that we have discussed earlier. In addition, we will discuss about the performance and the time reduced by the developers of their day to day repetitive work and how it has contributed to their work positively while increasing the efficiency and effectiveness of the projects they have undertaken.

7.4.1. Participants

Redot Pvt Ltd

Redot Pvt Ltd was founded in 2012 in Sri Lanka and is a blooming IT firm which delivers services to its clients in the fields of web applications and mobile applications virtually. It serves SME around the world and satisfies customers around the globe. Since most of the clients are from Singapore it has been already registered in Singapore as well. The services they deliver includes developing, designing, integrating and maintaining web services, mobile applications, Search engine optimizing, network and security and graphics.

The main participant of the evaluation of the project was a team of 10 developers from Redot Pvt Ltd in which most of them were from web application development background and working on web application projects by the time this evaluation was done. All the participants are having an experience ranging from 2-8 years in the industry and five developers out of ten are currently reading for their Master of Information technology having completed their first degree. The above-mentioned developers constantly expose themselves to the new updates and knowledge circulated in the industry and tries to incorporate these in their related work fields to make their work more user-friendly and convenient for the customers they serve.

Effro Pte Ltd

Effro is an on-demand talent marketplace which helps searchers who look for talents in different performances to find the required talent easily. It helps event planners find specific talents for every event, whether it's the conventional, emcees, dancers, models or singers or the unique tarot card readers, parkour performers or fire eaters. Effro is basically a registered company in Singapore and now serves it clients virtually allowing many people to access talents all over the world.

The software system which was generated using prototyping tool was tested by four testers from Effro Pte Ltd thus helped to evaluate it from different perspectives and allowed to bring different ideas and suggestions to make this project a success.

Ranomark Pvt Ltd

Ranomark International Pvt Ltd is a medium scale total IT solution providing company, working to help small, medium and large size businesses to be successful on the web. With expertise in all aspects of online business, they help to create a plan to make the web site a highly effective aspect of the business.

Ranomark has passionate designers, developers, and strategists with an insatiable thirst for building amazing products. They work with clients to develop new products, rebuild older applications and create prototypes. Their primary proficiencies are in HTML, PHP, CSS user experience design and mobile development.

A group of two developers and a tester was used to check the validity of the software system generated using our prototyping tool and their feedback on the same was taken into account when making the final adjustments to the prototyping too

7.4.2. Testing environment

Development Environment

The software systems generated using the prototyping tool was developed by the developers of both Redot Pvt and Ranomark Pvt Ltd. The two teams tried to generate different software systems and also tried to figure out the scenarios in which the systems were capable of delivering the desired output and also the scenarios which failed to deliver the desirable outputs. They tried using the systems to test different test cases in different projects so that it will help the system to be evaluated in different aspects.

Testing Environment

Testing was done in office work environment. And the pre-installation of the project (cloning GIT repository powered by bit bucket) was done before the beginning of the project and no language specific setups or modification was done since FES handles these tasks.

Testing was done by Effro Pte Ltd and they tested the adaptability, quality, installation, real time performance of the generated software which was generated using FES.

7.4.3. Test cases

Deferent test cases and methods were used while doing the research evaluation.

Time Saving

- 2 project teams were chosen from redot pvt ltd with same capabilities and knowledge.

- Each team was given set of tasks to perform
  - Task 1 - Setting up the project
    - Enabling version controlling
    - Initiation of the project
    - Setting up the dependency managers
    - Installing dependencies
    - Setting up environments
  - Task 2
    - Add a CRUD operation for user module
    - Write test cases for each endpoint
    - Follow industry accepted coding standards
  - Task 3
    - Modify the CRUD operations to support data manipulation with relationships while maintaining data integrity. Ex: user hasMany permissions
- Team 1 was equipped with Our Prototyping tool and the other team was following their day to day development tasks to achieve the same goals
- After 20 mins, team 1 was able to complete all 3 tasks and team 2 was still at setting up dependency managers.
- To complete all 3 tasks team 2 took more that 5+ hours

This test case shows us that the prototyping tool reduces the time taken for day to day and repetitive tasks by very high amount. This saves lot of time and money for the company.

Cost Saving

As mentioned above time taken for programming has reduced by 3000% which is a considerable amount and the man hours which was saved is around 10 since 2 developers were in a team.

Estimated cost saving for the company is around 14000LKR from those 3 tasks only.

Generated code quality check

Developers from ranomark pvt ltd was examining the generated source code and agreed that the generated source code follows industry standards. But it was generating some segments which wasn't required by the client. It had to remove manually by the developer.

Error free/ minimal error code

The prototyping tool generates the test cases for each of the endpoint which it generates. Developers tried to run these test cases and most of the time all of the test cases passed which suggest that the prototyping tool is generating code with minimal errors.

Handling data integrity

This tool is capable of generating model relationships and it had no issues in generating the required UI which involved multiple data models. It was capable of handling data integrity and run validation based on these data model dependencies

Data collection

Data was mainly collected through observations while the developers were engaged in developing the systems using the prototyping tool in the above mentioned developing environment. They were being mainly observed on the below categories such as

- the time they spent on coding the initial set up
- cost linked to the development

- employee engagement when using the systems developed using the prototyping tool
- efficiency and effectiveness of the projects carried out
- the extent of errors and malfunctions of the systems developed.

Personal Feedbacks

Personal feedbacks were taken from developers and testers from Redot Pvt Ltd, Ranomark Pvt Ltd and Effro Pte Ltd since they were the main developers who tested the systems which were generated using the prototyping tool. Their personal feedbacks also helped a lot to improve this project as they are also familiar with the latest technology used in the industry and also equipped with knowledge and the skills needed to develop the required sophisticated web applications and mobile applications.

7.4.4. Data analysis

Time spent on initial Set Up

It was observed that it would normally take a minimum of two weeks to finish off the initial set up of a project through manual coding depending on the requirements of a project. But by using the prototyping tool, the developers would be able to generate the initial Set Up in no time since that work is done by the prototyping tool automatically. Because of this, the time spent on the projects can be reduced and the developers can focus on other important functions of a project related to web development and mobile applications and save time for more complex problems arising in a project.

Cost related to the development

Since the manual coding requires a lot of time, resources and human capital when a project is being evaluated, a substantial amount of cost would be assigned to developers. But by using the prototyping tool the time taken can be reduced immensely thus allow the organizations to reduce the cost as well, because in a business time is considered as cost. And if the time can be reduced the cost also can be reduced. Since the developer can show the initial setup of the project to the end user during the first meeting itself it reduces the number of revisits to the end user and it helps to maintain a good relationship with the customer as well since it allows for both parties to be clear on the end deliverables from the first meeting itself.

Reduction of repetitive work

When a software system is developed, it involves lot of repetitive manual coding which can be different from project to project depending on the requirements of the project. The repetitive work will negatively affect the programmer as well since the same code has to be repeated over and over again. This will lead to monotonous work and it will directly affect the efficiency and the effectiveness of the project itself. Prototyping tool generates the CRUD operations of the generated system intelligently and as we have mentioned earlier, through the prototyping tool number of repetitive work has been reduced considerably and programmers were able to focus more on other important development and deployments tasks such as handling business logic, optimizing source code, optimizing performance and improving the UI generated of the project.

7.5. Aim

The main aim of this project is to reduce the time taken for repetitive development tasks while maintaining a good healthy code. Which ultimately lead to successful projects.

7.6. Objectives

- Generate the software source code same as a web application generator
- Automate the CRUD source code generating system
- Generate the flow of the system
- Support multiple front end and back end languages
- Support multiple databases
- Generate the source code which follows industry standards
- Generate test cases
- Allows project team members a cloud based prototyping solution

7.7. Drawbacks and limitations

Since the prototyping tool generates code automatically for the software required, it leaves less or no room for syntax errors and the codes generated are in par with the industry standards. It helps the developers to be more flexible and helps them to be more innovative and to go for that extra mile to satisfy their customers since it allows them feel more relaxed.

One drawback which was noticed from the developers was that it generates API Endpoints for all possible scenarios which sometimes is unnecessary. Developers had to remove these code segments manually to improve code quality further.

Another drawback which was noticed was that it cannot handle much complex scenarios since this project is an initial foundation a prototyping tool in which improvements can be laid on going forwards.

7.8. Summery

This chapter fully discussed about the evaluation and the testing of the system according to the aim and objectives defined. Results of this evaluation were given to managing director at Redot Pvt Ltd. Next chapter will discuss on the conclusion and further works of the project.

Chapter 8

8. Conclusion

8.1. Introduction

Overall achievement of the Service Oriented Fast Prototyping based on requirement definition schema is completed and successful. Following are the tasks which was targeted by the system.

- Automated CRUD generation
- Automated application flow generation
- Automated user input validation for the generated code
- Automated application UI elements
- Automated application routing mechanisms
- Supporting multi-platforms and multiple databases
- Generate multi-language code
- Generation of testable and reusable code
- Allow cloud based prototype generation

8.2. Conclusion

Building a Service Oriented Fast Prototyping based on requirement definition schema was achieved successfully. By studying how the users of the system is interacted we can see that it can reduce their project starting up time greatly.

I'm pleased to mention that I've gone through various validation mechanisms, open-source code generation tools, software code best practices. Latest technologies such as angular4. As a result, I'm now an experienced angular4 developer and getting a high demand at my work place as well.

By doing this research, my coding skills improved since the code which was generated by the system had to match the industry standards and developer friendly. I've learnt about open source testing tools which was used to test the generated code.

When it comes to the prototyping tool, it was a really useful project for me as a developer and for other developers as well. And this prototyping tool quickly became a popular tool among my fellow colleagues. Three systems are already built by using this prototyping tool, and 1 project was deployed recently which is http://www.effro.com .

As a conclusion, Service Oriented Fast Prototyping based on requirement definition schema was a successful research and achieved its goals. Reducing around 1 – 4 weeks of work from the development phase.

8.3. Future works

Even though the prototyping tool is generating the code which developers needed, developer should have some knowledge of programming and write the code in JSON format as non-technical users require some learning to use this tool. We can introduce a Designing tool for this project which will generate the JSON schema required by the prototyping tool. which will make the FES usable for even by a non-technical person. And this will highly be useful for Business Analysists, Project managers as well as system designers. And can be introduced for generating fully functional small-scale applications.

As I mentioned above this prototyping tool requires only 2 JSON schemas and it uses a simplified language to define the system. We can make it more user friendly by introducing voice recognition and low-level API which will generate the required JSON schema for the client requirement. And this will lead to an application which can listen to user's requirement and generate a prototype based on that.

Another improvement planned on this is that the interactive designer similar to what we have found during the evaluation. This will allow users to design their systems more easily.

8.4. Summery

This chapter provided a conclusion of overall solution achieved by the Research project named Service Oriented Fast Prototyping code generation based on requirement definition schema done at faculty of IT at University of Moratuwa and further work as an enhancement of the current project.

References

[1] S. Mbarki and M. Erramdani, “Toward automatic generation of mvc2 web applications,” *ResearchGate*.

[2] S. Lazetic, D. Savic, S. Vlajic, and S. Lazarevic, “A generator of MVC-based web applications,” *World Comput. Sci. Inf. Technol. J. WCSIT*, vol. 2, no. 4, pp. 147–156, 2012.

[3] M. L. Bernardi, G. A. Di Lucca, and D. Distante, “A model-driven approach for the fast prototyping of web applications,” in *Web Systems Evolution (WSE), 2011 13th IEEE International Symposium on*, 2011, pp. 65–74.

[4] R. Cheung, “XFlash–a web application design framework with model-driven methodology,” *Int. J. U- E-Serv. Sci. Technol.*, vol. 1, no. 1, pp. 47–54, 2008.

[5] P. Vuorimaa, M. Laine, E. Litvinova, and D. Shestakov, “Leveraging declarative languages in web application development,” *World Wide Web*, vol. 19, no. 4, pp. 519–543, Jul. 2016.

[6] A. Bulajic, S. Sambasivam, and R. Stojic, “An effective development environment setup for system and application software,” *Issues Informing Sci. Inf. Technol.*, vol. 10, pp. 37–66, 2013.

[7] “Auto PHP Codeigniter CRUD | Grocery CRUD.” [Online]. Available: http://www.grocerycrud.com/. [Accessed: 28-Aug-2016].

[8] “Angular Full-Stack.” [Online]. Available: http://angular-fullstack.github.io/generator-angular-fullstack/. [Accessed: 28-Aug-2016].

[9] “Rapid mobile app and web application development platform.” [Online]. Available: http://www.webratio.com/site/content/en/home. [Accessed: 28-Aug-2016].

[10] “Node.js.” [Online]. Available: https://nodejs.org/en/. [Accessed: 18-Dec-2016].

[11] “Express 4.x - API Reference.” [Online]. Available: https://expressjs.com/en/4x/api.html. [Accessed: 08-Sep-2016].

[12] “JSON Schema.” [Online]. Available: http://json-schema.org/. [Accessed: 18-Dec-2016].

[13] “Angular.” [Online]. Available: https://angular.io/. [Accessed: 17-Dec-2016].

[14] “Sequelize | The Node.js / io.js ORM for PostgreSQL, MySQL, SQLite and MSSQL.” [Online]. Available: http://docs.sequelizejs.com/en/v3/. [Accessed: 21-Apr-2017].

[15] “Mocha - the fun, simple, flexible JavaScript test framework.” [Online]. Available: https://mochajs.org/. [Accessed: 21-Apr-2017].

[16] “Chai.” [Online]. Available: http://chaijs.com/. [Accessed: 21-Apr-2017].

[17] Atlassian, “Bitbucket | The Git solution for professional teams,” *Bitbucket*. [Online]. Available: https://bitbucket.org. [Accessed: 03-May-2017].

[18] “Visual Studio Code - Code Editing. Redefined.” [Online]. Available: http://code.visualstudio.com/. [Accessed: 03-May-2017].

[19] phpMyAdmin contributors, “phpMyAdmin,” *phpMyAdmin*. [Online]. Available: https://www.phpmyadmin.net/. [Accessed: 03-May-2017].

[20] “WAMP, MAMP and LAMP Stack : Softaculous AMPPS.” [Online]. Available: http://www.ampps.com/. [Accessed: 03-May-2017].

[21] Mobatek, “MobaXterm free Xserver and tabbed SSH client for Windows.” [Online]. Available: http://mobaxterm.mobatek.net/. [Accessed: 03-May-2017].

[22] “Elastic Compute Cloud (EC2) – Cloud Server & Hosting – AWS,” *Amazon Web Services, Inc.* [Online]. Available: //aws.amazon.com/ec2/. [Accessed: 03-May-2017].

[23] “AWS Elastic Beanstalk – Deploy Web Applications,” *Amazon Web Services, Inc.* [Online]. Available: //aws.amazon.com/elasticbeanstalk/. [Accessed: 03-May-2017].

[24] “angular-ui/ui-router,” *GitHub*. [Online]. Available: https://github.com/angular-ui/ui-router. [Accessed: 21-Apr-2017].

[25] “What is RESTful API? - Definition from WhatIs.com,” *SearchCloudStorage*. [Online]. Available: http://searchcloudstorage.techtarget.com/definition/RESTful-API. [Accessed: 28-Mar-2017].

[26] “gulpjs/gulp,” *GitHub*. [Online]. Available: https://github.com/gulpjs/gulp. [Accessed: 18-Dec-2016].

[27] “Digital Product Design, Workflow & Collaboration,” *InVision*. [Online]. Available: https://www.invisionapp.com/. [Accessed: 03-May-2017].

[28] “Proto.io - Prototypes that feel real.” [Online]. Available: https://proto.io. [Accessed: 03-May-2017].

[29] “Moqups · online mockups made simple.” [Online]. Available: https://moqups.com. [Accessed: 03-May-2017].

Appendix A

Interfaces of the generated prototypes

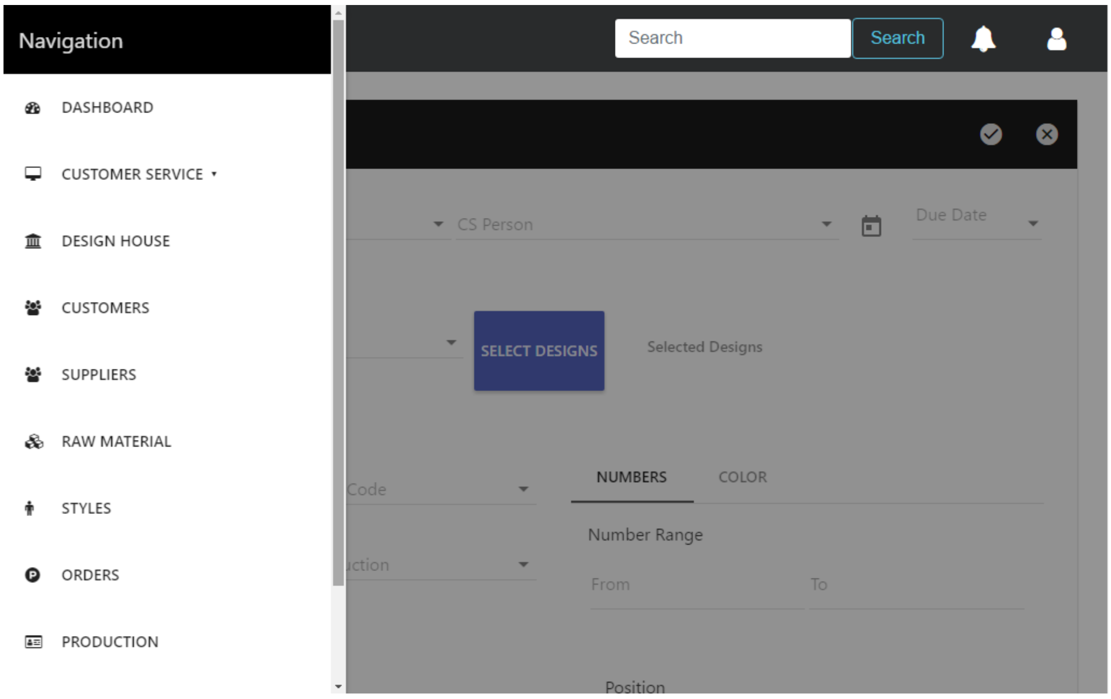


Figure Dashboard UI view

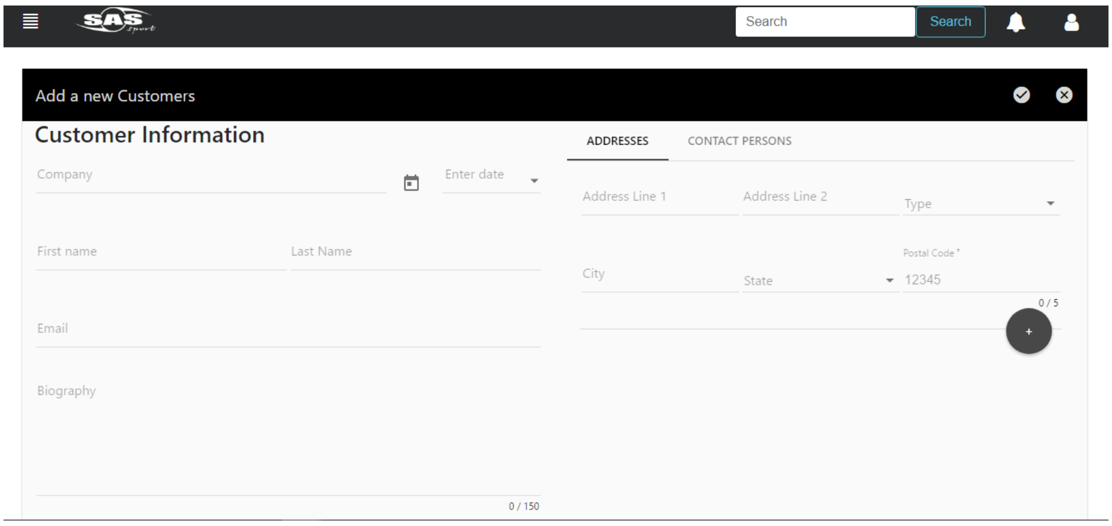
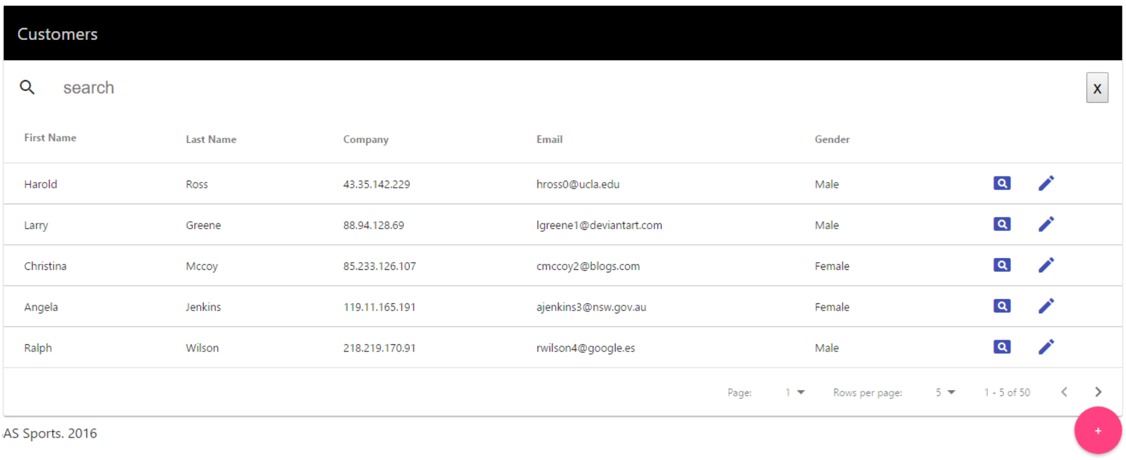


Figure UI view of a CRUD operation



Customers

search

First Name	Last Name	Company	Email	Gender
Harold	Ross	43.35.142.229	hross0@ucla.edu	Male
Larry	Greene	88.94.128.69	lgreene1@deviantart.com	Male
Christina	Mccoy	85.233.126.107	cmccoy2@blogs.com	Female
Angela	Jenkins	119.11.165.191	ajenkins3@nsw.gov.au	Female
Ralph	Wilson	218.219.170.91	rwilson4@google.es	Male

Page: 1 Rows per page: 5 1 - 5 of 50

AS Sports. 2016

Figure UI components generated for customer listing

Appendix B

Source code Segments

```
import { FileManager } from './file.manager';
import { angularjs } from '../language-drivers/angularjs';

export class FES {


    constructor() {
        this.f = new FileManager();
    }

    init() {
        this.initAppGenerationFlow();
    }


    initAppGenerationFlow() {
        this.getFileList();

        if (this.schemaFiles.indexOf('flow.schema.json') == -1) {
            throw new Error("flow.schema.json not found in schema directory");
        }

        if (this.schemaFiles.indexOf('models.schema.json') >= 0) {
            this.parseModels();
        }
        if (this.schemaFiles.indexOf('models.schema.json') >= 0) {
            this.parseModels();
        }

        this.parseFlow();


        var frontEnd = new angularjs(this.flowJSON, this.modelsJSON);

    }

    getFileList() {
        this.schemaFiles = this.f.getList('schema');
    }

    parseModels() {
        this.modelsJSON = this.f.getJsonFileContent('schema/models.schema.json');
        this.modelsJSON.relationships = [];

        for (var model of this.modelsJSON.models) {
```

Figure Application initiation code

```
var path = require('path');
var fs = require('fs-extra');
var ejs = require('ejs');

export class FileManager {

    constructor() {
        this.checkForPreviousBuilds();
    }

    checkForPreviousBuilds() {
        try {
            console.log(this.fileMnager.getFileContent('dist/front-
end/package.json'));
            this.doFrontendReplace = true;
        } catch (e) {
            this.doFrontendReplace = false;
            console.log("No Previous Fronend Builds");
        }
        try {
            this.fileMnager.getFileContent('dist/back-end/package.json');
            this.doBackendReplace = true;
        } catch (e) {
            this.doBackendReplace = false;
            console.log("No Previous Backend Builds");
        }
    }

    getList(location) {
        return fs.readdirSync(path.resolve(__dirname, '../../' + location));
    }

    getFileContent(location) {
        return fs.readFileSync(path.resolve(__dirname, '../../' + location),
'utf8');
    }

    getJsonFileContent(location) {
        return JSON.parse(this.getFileContent(location));
    }

    rename(oldname, newname){
        fs.renameSync(path.resolve(__dirname, '../../' + oldname),
path.resolve(__dirname, '../../' + newname));
    }
```

Figure File Manager Code

```
import { FileManager } from '../../lib/file.manager';
import { ViewGenerator } from './view.generator';

export class angularjs {

    constructor(flow, models) {

        this.flow = flow;
        this.models = models;

        this.app = {
            modules: []
        };

        this.topLevelModules = [];

        this.viewGenerator = new ViewGenerator(this.models);
        this.fileMnager = new FileManager();
        if (!this.fileMnager.doFrontendReplace) {
            this.copySeed();
        }

        this.generateModuleFiles();


    }

    copySeed() {
        this.fileMnager.copyFiles('src/language-drivers/angularjs/seed/project/',
'dist/front-end/');
    }

    generateModuleFiles() {
        this.generateModule(this.flow);

        //this.fileMnager.delete('dist/front-end/src/client/app/modules/');
        //write module files
        for (let m of this.app.modules) {
            if (m.id != 'main') {
                for (let f of m.files) {
                    this.fileMnager.createOrOverwrite('dist/front-
end/src/client/app/modules/' + m.path.replace(m.id, '') + '/' + m.id + '/' +
f.file, f.content);
                }
            }
```

Figure Language Driver for Angular (sample code)