**Anonymous Builders Customer/Job Data Application—Project Plan**

1. **Introduction:** Anonymous Builders (AB), provider of residential construction services, seeks a complete replacement of their existing data system, used to record details of customers and the jobs associated with them. The company currently relies on a process that is initiated by hand-filled paper forms and then transcribed into electronic spreadsheets. AB’s objective is to eliminate the need for any paper documentation in the initial job assessment process, secure storage of all customer, job, and job site photo data off site from the office, and make data accessible only to authorized personnel. The interface must be easily utilized by non-technical personnel. Time constraint is the completion of the project within 16 weeks of the start date.
2. **Project Organization:** The project team leader is Josh Belcher; he will be responsible for setting up the AWS hosting, database storage, configuring AWS services, writing custom Lambda scripts, and using the AWS SDK to connect the web pages to the API, while ensuring the team is communicating and that the different work areas are integrating properly. Another team member is Logan Manlief; he is responsible for setting up the frameworks and developing the frontend code that will create the interactive GUI for the user. The last team member is Tristan Barnett, he is responsible for layout of the web app’s pages and configuration/layout of their various elements.
3. **Risk Analysis:** The risk analysis content can be found in a separate document, *Risk\_Assessment.docx*.
4. **Hardware and Software Requirements:**

*Hardware requirements:*

* AB’s existing PCs
* The personal mobile devices of AB’s employees (various smartphones, tablets, etc.)
* AB intends to provide basic Android tablets to employee who do not already own a suitable Chrome or Firefox compatible device
* Off-site cloud servers provided by third-party hosting service, *Amazon Web Services*

*Proposed software requirements:*

* AWS services
* DynamoDB database
* IAM User Policies
* S3 Storage
* Lambda
* API Gateway
* Cognito Authentication
* Firefox and Chrome browser installations
* Browser UI
* Create using raw HTML, CSS, JavaScript, plus elements of jQuery and Bootstrap
  + Custom Node.js scripts that perform the data retrieval and processing between the database and the API

*Languages used:* We propose to develop the browser UI using the aforementioned trio of web languages. JavaScript with Node.js runtime has been chosen as the development language for our API.

1. Work Breakdown Structure (WBS):

Task that have been accomplished at this point in the project will be tagged by the team member responsible (in red text for visibility). Items only partially accomplished will be noted as “PARTIAL.”

**Project Name: Anonymous Builders Client/Job Data Application**

1. Front-End View
   1. Gather Requirements *(Inputs)*

1.1.1 Technical Specifications *(Josh)*

1.1.2 User Requirements *(Josh)*

1.1.3 Reporting Requirements *(Josh)*

1.2 Research browser-based, web GUI options *(Inputs)*

1.2.1 Scripting/markup language selection *(Logan)*

1.2.2 Possible framework benefits/challenges *(Logan)*

1.2.3 Architecture patterns *(Josh)*

1.3 Select Technical Framework and associate with Design Document *(Outputs)*

1.3.1 Evaluate options researched against requirements *(Logan)*

1.3.2 Evaluate time to develop *(Josh)*

1.3.3 Make decisions *(Josh, Logan, Tristan)*

1.4 Build page/form elements markup *(Outputs)*

1.4.1 Design Elements *(Logan, Tristan)*

1.4.2 Design Overall Layout *(Logan)*

1.4.3 Design Content of Elements *(Logan, Tristan)*

1.4.4 Create “Login” Page *(Tristan)*

1.4.5 Create “Create New Client” page *(Logan, Josh)*

1.4.6 Create “Find Client & Update Records” *(Logan, Josh)*

1.4.7 Add Job retrieval feature to “Find Client & Update Records” page

1.4.8 Add ability to update Client and Job records to

“Find Client & Update Records” page

1.4.9 Add Client and Job deletion feature to “Find Client & Update Records” page

1.5 Code interactions *(Outputs)*

1.5.1 Form submission/retrieval of data *(Logan, Josh PARTIAL)*

1.5.2 Login and session authentication (AWS Cognito) *(Josh)*

1.5.3 Integrate front-end and back-end *(Logan, Josh PARTIAL)*

1.6 Styling *(Outputs)*

1.6.1 CSS styles *(Logan, Tristan PARTIAL)*

1.6.2 Company logo/other graphics

1.7 Testing *(Outputs)*

1.8 Draft Corrections *(Outputs)*

1.9 Corrections Testing *(Outputs)*

1. Back-End Controller

2.1 Research back-end options

2.1.1 Languages for API script/runtime on web server *(Josh, Logan)*

2.1.2 AWS Services *(Josh, Logan)*

2.2 Code script to handle database authentication/queries *(Outputs)*

2.2.1 Login for users *(Josh)*

2.2.2 Client record creation *(Josh)*

2.2.3 Client record edit/deletion (also deletes associated jobs)

2.2.4 Job record creation

2.2.5 Job record edit/deletion

2.3 Testing *(Outputs)*

2.4 Draft Corrections *(Outputs)*

2.5 Correction Testing *(Outputs)*

1. Back-end database
   1. Research AWS database options *(Inputs) (Josh)*
   2. Setup AWS database (DynamoDB) *(Outputs)*

3.2.1 Create database *(Josh)*

3.2.2 Create Customers, and Job tables *(Josh)*

* + 1. Populate with test record data

3.3 Implement any needed AWS features *(Outputs)*

3.3.1 Lambda *(Josh)*

3.3.2 API Gateway *(Josh)*

* + 1. AWS Cognito and IAM user roles/policies *(Josh)*
  1. Testing *(Outputs)*
  2. Draft Corrections *(Outputs)*
  3. Corrections Testing *(Outputs)*

1. Project Management Deliverables *(Outputs)*
   1. Requirements Document *(Josh, Logan, Tristan)*

4.2 Feasibility Study *(Josh, Logan, Tristan)*

4.3 Project Plan *(Josh, Logan, Tristan)*

4.4 Risk Assessment *(Josh, Logan, Tristan)*

4.5 Design Document *(Josh, Logan, Tristan)*

4.6 User Documentation

4.7 Team Evaluation

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Team | M01 | M02 | M03 | M04 | M05 | M06 | M07 | M08 |
| View  Dev. | Josh,  Tristan | Research front-end web GUI options, including languages, frameworks, and architecture patterns | | | | Design layout of GUI, form elements, data input collection, event handling | Code Implmt.  &  Testing | Code GUI styling &  Testing |  |
|  |  | | | | |  |  |  |  |
| Controller  Dev. | Josh, Logan | Research back-end options, including AWS API Gateway, AWS Lambda, and associated runtimes/languages | | | | Design authorization & interactions between API, Lambda calls, database, and GUI | Code Implmt.  &  Testing | Code picture upload feature  &  Testing |  |
|  |  | | | | |  |  |  |  |
| Model  Dev. | Josh,  Logan | Research options for AWS hosting, including AWS RDS database selection and implementation | | | | Design schema for database tables and relationships | Code Implmt.  &  Testing | Further Testing |  |
| Proj.  Mgmt.  Deliverables | All | Req.  Doc. | Feasibility  Study | Proj.  Plan | Risk  Assess. | Design  Doc. | Software Project  Implementation  Draft Final | | User Docs. &  Team Eval. |

**Diagram, engineering drawing

Description automatically generated**

1. *Process Flow Table and Diagram (above)*
2. **Project Schedule: *see associated Microsoft Project file***
3. **Monitoring and Reporting Mechanisms:** The team will use the following means to communicate progress, problems, and general collaboration. Team leader will provide progress reports to product owner as required.

* **Team Discord server**
* **GitHub commit messages**
* **Diligent code commenting**
* **Weekly Zoom conferences**

1. **Appendix: *see associated Microsoft Project file***