RECEIPT KEEPER – IMPLEMENTING RESTFUL API USING LOOPBACK FRAMEWORK

C. Choi, H. Nahhal, N.Collins, S. Kim

Advisor D., Dvorski

Mobile Solution Development,

School of Engineering & Information Technology

Conestoga College, ON, Canada

***Abstract- One type of document that is frequently used in daily life is receipts. Receipts can be used for the tracking of personal spending or should be preserved for tax reporting or refund purposes. In the professional area, employees’ spending during the performance of their tasks should be reported to the business account.***

***Add how we will solve the problem.[Joe]***

I. INTRODUCTION

Receipt Keeper will allow users to track how much money they spent in a certain category over a specific number of months, as well as start a group which others can be invited to join, and this will allow group admins to see reports for the group. Receipt Keeper is built to work on Android devices and also includes a web app. The mobile application will make it easier for people to manage categories, tags and reports while having the convenience of scanning receipts wherever they are. The web application provides the list of receipts and shows graphs to visualize users’ spending.

*A. Challenges and objectives*

The biggest challenge was to make a working product within limited time and limited resources.

*1) OCR engine to recognize numbers and characters on user’s receipt:* due to lack of knowledge on OCR, open source products were considered which include OpenCV, Tesseract, etc.

2) *Providing Rest API for scalability:* A lot of different devices should be connectable to the server such as web application, Android, iPhone, Tablet, even customized devices. Implementing a Server which is connectable with API is another challenge.

3) *Android Application using Rest API to communicate with Server:* To implement a working product within the time available, the team is focusing on implementing

II. METHODOLOGIE

*A. Development Methodologies & Team Rules*

1. Agile Unified Process:
2. Coding rule

Coding rule should be minimized to every member can follow the rule and should be updated after members are accustomed to the rule. Coding rule also includes comment rule for Git, JavaScript, and Android.

*B. Source Control & Change Management*

1. GitHub

Three GitHub projects are created, one each for managing document, web server & app - android.

1. Rule

Direct commit was discouraged, but making pull request was encouraged for peer review and made a rule for source code management.

1. Sharing Information

To share the information such as coding rule, link information, and minutes, Wiki page at GitHub was used.

*C. Issue Tracking*

GitHub’s issue tracking function was used to simplify the managing. Although it has not many functionality compared other issue tracking such as Jira, it can be connected with GitHub seamlessly.

*D. Backend as a Service (BAAS)*

LoopBack framework was used and will be stated in detail later.

*E. Deployment*

For supporting interoperation during the development, it is important to set up the server environment early so that every member can use it. HEROKU provides a free account for prototyping. However, one of drawbacks of a free account is server will go into sleep mode after 30 minutes of inactivity, so it needs to be activated before using the server, but it is not a big problem for prototyping. Moreover, HEROKU supports buildpacks for deploying LoopBack app, so after uploading source code with Git to the HEROKU environment, it will be compiled and deployed automatically.

To avoid the problem of cracking the operation of the server, pipeline features at HEROKU was used. After setting up, every pull request at project’s GitHub will trigger a build at HEROKU, it will pull from GitHub and build automatically. If the source code submitted is stable, it will be promoted to the production stage manually. This server will be a common server shared between members. -> Review App -> Staging -> Production.

REVIEW APPS

STAGING APPS

PRODUCTOTION APPS

Fig. 1. HEROKU Pipeline

*F. Data as a Service (DAAS)*

Although if team is getting bigger and bigger, setting up an internal customized using a container such as Docker, is needed. However, to reduce time setting up the environment, it was decided to use cloud database service. Because LoopBack does not support cloud database service for free, it was decided to use another database service. Among the providers, mLab provides Database-as-a Service with various types of plans and features for MongoDB. Among the plans, the Sandbox plan provides 500MB of space free which is enough for development and prototyping, although it is slow compared to non-free plans.

*E. Sharing Issues and Communication*

Because team members work in separate places, although one day is allocated for Capstone project lab, it is necessary to set up communication channels other than phone. Slack is a famous messaging app for professional use. Channels were set up for the subject of discussion and have been used for the discussion of interoperation or sharing knowledge.

Additionally, to share the information such as meeting minutes, coding rules and documentation, GitHub’s Wiki was used because managing many tools or sites can be another burden for the team, GitHub was used as a hub to share the outcome of the project.

III. ARCHITECTURE DECISIONSITS

ADD Use case diagram

*A. Architectural Driver*

1. Quality Attributes
2. Scalability: Extensible to other kind of client
3. Flexibility: Easy to change API
4. Performance: Quick to search
5. Security:

ADD detailed rationale (add QAS)

1. Architectural Decisions
2. REST API
3. NO SQL for the speed of search and Node.js
4. Security: CORS in middleware

IV. TECHNOLOGIES

Because one of the application’s main features is to help a user entering receipts by using Optical Character Recognition (OCR) technologies, OCR is main technology that is used.

1. Optical Character Recognition (OCR)

Due to lack of knowledge of OCR, team decided to use open source for character recognition. OpenCV and Tesseract are the products considered. OpenCV is a famous computer vision open source product, and Tesseract is a famous OCR product. There are a lot of open source project subsidiaries based on Tesseract. Although OpenCV was not used for this project, it can be used for pre-processing receipt images such as contrast enhancement, auto tilting of image to enhance recognition rate for OCR engine in a future project.

1. LoopBack

Another main technology is a server to support REST API. Various platforms are considered, but due to lack of team’s skills and to minimize time to implement the server, LoopBack is chosen finally. Table1 is the comparison of features between BAAS (Back end As a Service) solutions.

LoopBack is an open source Node application framework based on Express which is supported by IBM and StrongLoop (Acquired by IBM on September, 2015). It makes it easy to create Rest API platforms and managing REST API with provided tools, such as creating new API or apply access control list (ACL). Moreover, it has Built-in API explorer to test API which is useful to test the server operation before implementing web client or mobile application.

LoopBack supports a Yeoman-based scaffolding tool for generating new project skeletons, so it takes a few minutes to make skeletons for the project.

LoopBack helps with that further by providing a model generator for building your models quickly. Entity relationships are also very well supported. The security model is also complex with user roles, principals and ACLs

Authentication and authorization are important tasks for implementing a server, but it was easy to setup authentication and authorization by using LoopBack’s built-in role based access controls. Open Standard for Authorization (OAuth) user and registration models is baked in the Loopback, so it is also easy to embed third party logins such as Google and Facebook.

StrongLoop Arc is a graphical tool for building, deploying, managing and monitoring LoopBack applications and APIs.

Developers have a choice of various database connectors for all major SQL databases and MongoDB. LoopBack helps with that further by providing a handy model generator for building your classes quickly. These can be validated with the built-in validation methods. Entity relationships are very well supported. The security model is complex with user roles, principals and ACLs.

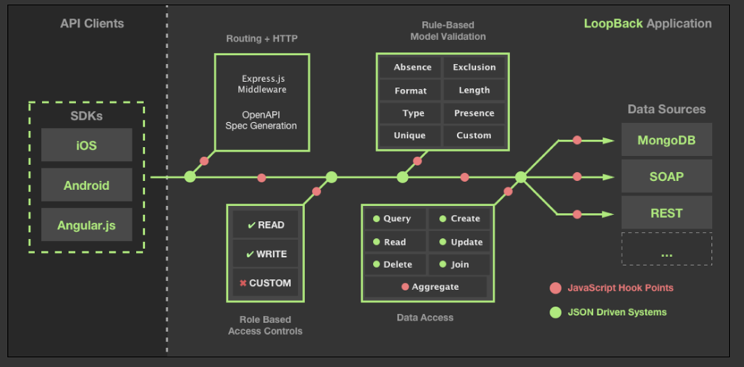


Fig. 2. LoopBack Application

TABLE I. COMPARISON OF BAAS (BACKEND AS A SERVICE)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **LoopBack** | **Express** | **Sails** | **Restify** | **Meteor** |
| **Type** | API framework | HTTP server library | Web MVC framework | REST HTTP library | Full-stack JavaScript app platform |
| **Top Features** | Enterprise connectivity, API Explorer, generators, client SDKs, websocket microservices | HTTP routing, middleware | Rails familiarity, MVC | Simplicity, REST routing | Universal JavaScript, reactive rendering, websocket microservices |
| **Suitable For** | Web apps, APIs | Simple web apps | Web apps, APIs | Simple REST APIs | Web apps |
| **Github Stars** | 5k | 19k | 10k | 3k | 28k |
| **Support** | StrongLoop | StrongLoop | N/A | N/A | Meteor Development Group |
| **Pure Node runtime** | Yes | Yes | Yes | Yes | No |
| **Client SDKs** | Angular, Browser, Node.js, iOS, Android, Xamarin | N/A | None | None | JavaScript, Cordova for iOS and Android, React, AngularJS |
| **Export API Definition** | Yes | With strong-remoting | None | None | With meteor-rest |
| **Tools** | Visual API composer, Explorer, CLI code generators | CLI app generator | Yeoman generator | Yeoman generator | CLI tool |
| **Visual API composition** | Yes | No | Yes | Yes | Yes |
| **StrongLoop Arc Build & Deploy, Monitoring, Profiling** | Yes | Yes |  |  | Proprietary package system and repository, npm |
| **Extensions** | Push, File Storage, Passport, OAuth 2.0, Express Middleware | Express / Connect Middleware | In-memory, File, PostgreSQL, MySQL, MongoDB | None | MongoDB, MySQL and PostgreSQL via 3rd-party packages |
| **Data sources** | In-memory/file, MongoDB, MySQL, Oracle, PostgreSQL, SQL Server, ATG, Email, REST, SOAP | None | No | No | Basic allow/deny |
| **ACLs** | Yes | No |  |  |  |

V. OPEN SOURCES

A lot of open sources are help to implement features within short time, but a lot of points should be considered beforehand such as License, sustainability, and securities.

1. Server

|  |  |  |
| --- | --- | --- |
| **Open Source** | **License** | **Function** |
| **bunyan: ^1.8.1,** | **Apache 2.0** |  |
| **compression: ^1.0.3,** |  |  |
| **cors: ^2.5.2,** |  |  |
| **errorhandler: ^1.1.1,** |  |  |
| **express-bunyan-logger: ^1.3.0,** |  |  |
| **helmet: ^1.3.0,** |  |  |
| **loopback: ^2.22.0,** |  |  |
| **loopback-boot: ^2.6.5,** |  |  |
| **loopback-component-explorer: ^2.4.0,** |  |  |
| **loopback-component-storage: ^1.5.0,** |  |  |
| **loopback-connector-mongodb: ^1.15.2,** |  |  |
| **loopback-datasource-juggler: ^2.39.0,** |  |  |
| **loopback-ds-timestamp-mixin: ^3.2.4,** |  |  |
| **morgan: ^1.7.0,** |  |  |
| **serve-favicon: ^2.0.1** |  |  |

1. Web Client

|  |  |  |
| --- | --- | --- |
| **json3: ~3.3.1,** |  |  |
| **font-awesome: 4.3.0,** |  |  |
| **oclazyload: ~0.5.2,** |  |  |
| **angular-loading-bar: ~0.7.0,** |  |  |
| **angular-chart.js: ~0.5.2,** |  |  |
| **angular-ui-router: ^0.3.0,** |  |  |
| **angular-resource: ^1.5.6,** |  |  |
| **bootstrap: ^3.3.6,** |  |  |
| **eonasdan-bootstrap-datetimepicker: ^4.17.37** |  |  |

1. Android Mobile App

|  |  |  |
| --- | --- | --- |
| **com.rmtheis:tess-two:5.4.0** |  |  |
| **com.strongloop:loopback-sdk-android:1.5.+** |  |  |
| **com.github.bumptech.**  **glide:glide:3.6.0** |  |  |
| **com.rmtheis:tess-two:5.4.0** |  |  |
| **Tesseract: https://github.com/tesseract-ocr/tesseract** |  |  |

V. APPLICATON PROGRAM INTERFACES (API)

Use either SI (MKS) or CGS as primary units. (SI units are encouraged.) English units may be used as secondary units (in parentheses). An exception would be the use of English units as identifiers in trade, such as “3.5-inch disk drive.”

*A. Application Layer*

1. *Routes*
2. *Middleware*
3. *Errors*
4. *Formatters*

*B. Domain Layer*

*C. Service Layer*

1. *Data Access*
2. *JWT (JSON Web Token)*
3. *Authentication*
4. *ACL(Access Control List)*

*D. Foundation Layer*

1. *Database*

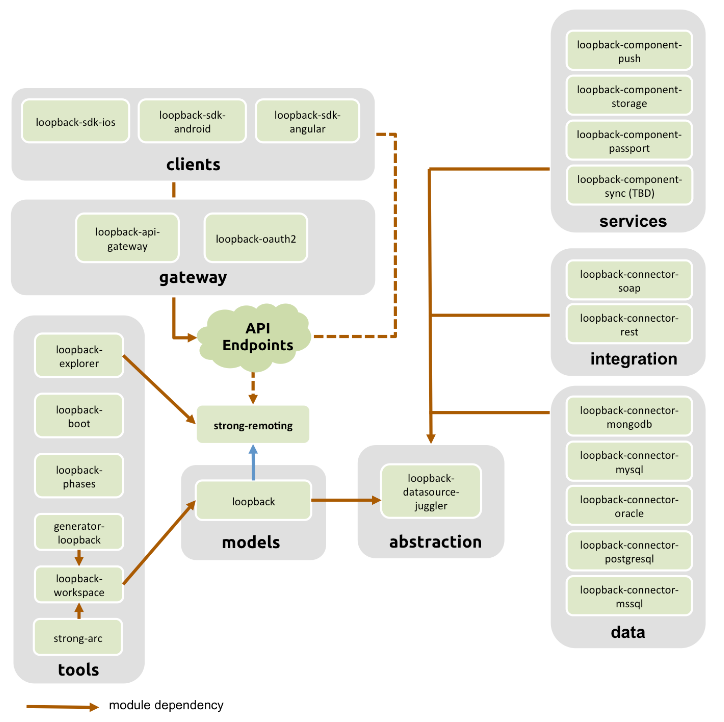


Fig. 2. LoopBack Modules

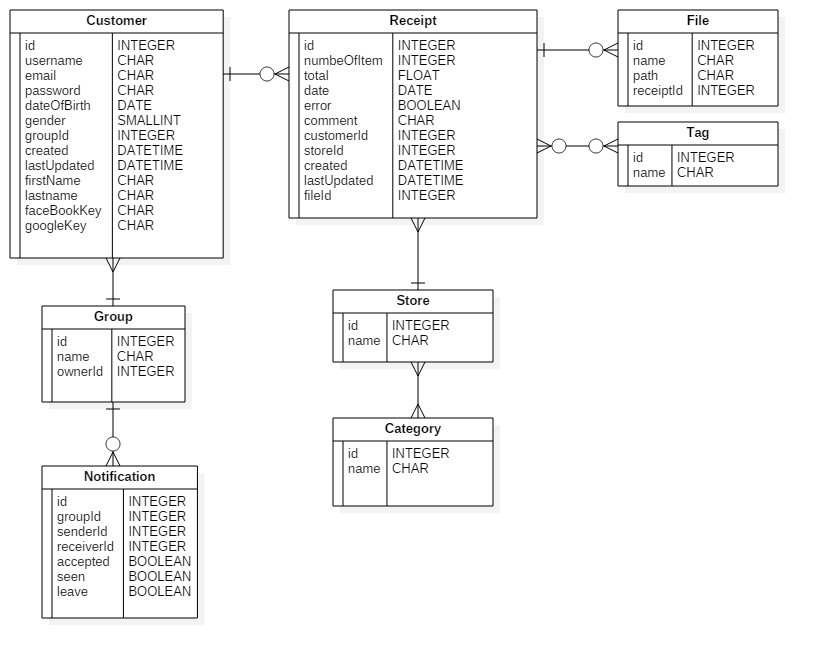


Fig. 3. Entity Diagram

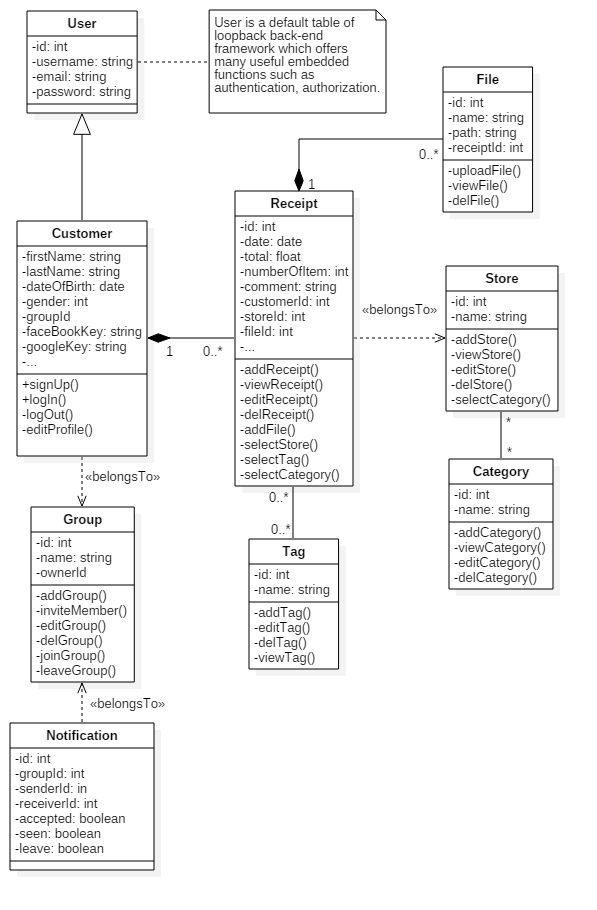


Fig. 4. Class Diagram

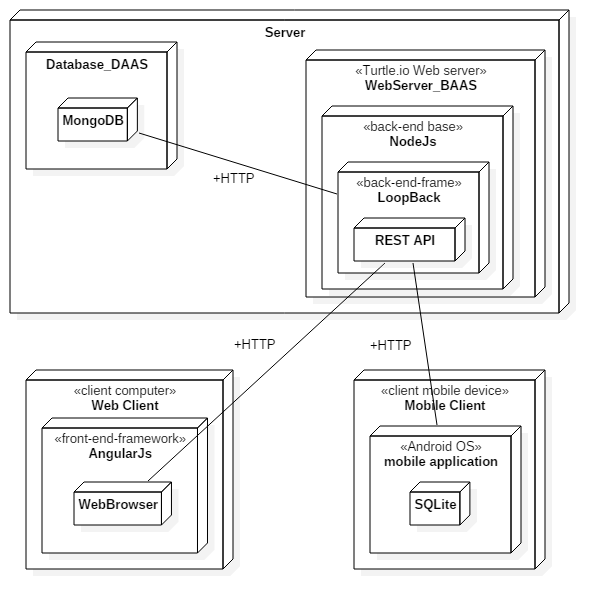


Fig. 5. Deployment Diagram

VI. CONCLUSIONS

By using the Loopback framework, it is easy and quick to implement Rest API for scalability. It is also fast because is based on NodeJS and easy to implement a single page application with the help of AngularJS

VII. ACKNOWLEDGMENT

REFERENCES

[1] android-ocr : https://github.com/rmtheis/android-ocr

[2] HEROKU, [*https://www.heroku.com/pricing*](https://www.heroku.com/pricing)

[3] LoopBack, [*https://docs.strongloop.com/display/public/LB/LoopBack*](https://docs.strongloop.com/display/public/LB/LoopBack)

[4] OCR, *https://en.wikipedia.org/wiki/Optical\_character\_recognition*

[5] OpenCV: *https://en.wikipedia.org/wiki/OpenCV*

[6] Tesseract OCR: [*https://github.com/tesseract-ocr/tesseract*](https://github.com/tesseract-ocr/tesseract)

Glossary

***A***

ACL:

Access control list, a list associated with an object that identifies all the subjects that can access the object and their access rights. See Authentication, authorization, and permissions.

*B*

Buildpacks:

Buildpacks are scripts that are run when the app is deployed. They are used to install dependencies for the app and configure the environment.

.

*C*

*D*

Data source:

A data source connects with specific database or other back-end system using a connector.

*E*

*F*

*G*

*H*

*I*

*J*

*K*

*L*

*M*

*N*

*O*

OCR (Optical character recognition):

Optical character recognition (optical character reader, OCR) is the mechanical or electronic conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo (for example the text on signs and billboards in a landscape photo) or from subtitle text superimposed on an image (for example from a television broadcast)

OpenCV:

OpenCV (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision, originally developed by Intel's research center in Nizhny Novgorod (Russia), later supported by Willow Garage and now maintained by Itseez.[1] The library is cross-platform and free for use under the open-source BSD license.

*P*

*Q*

*R*

*S*

*T*

*U*

*V*

*W*

*X*

*Y*

*Z*