

Step 1:

Custom Development Strategy.

The best design strategy for our system would probably have to be custom development strategy. By allowing our company to use the custom development system, we are able to build a new system specifically tailored to the business needs. There are not many websites similar to ours so our needs may be different. Using the custom development will allow us to be more flexible and can allow us to adapt better than other strategies. Having the ability to change things easier to adapt what the consumer needs will help us a lot. Custom development also allows the company to learn, develop, and grow by doing everything in house. Allowing employees to build on their skills within the company can be great for them as well. It is always a good idea to help employees expand their skills. The custom strategy also works well considering the time frame is flexible.

Outsourcing Strategy.

I also think that our business could use the outsourcing strategy a little bit as well. As the business is small and just starting, it may lack employees with the right experience. Not every part of this system would need to be outsourced, but some aspects of the system would be a lot easier to handle and maintain since the business is small. Other outside companies could be a lot of extra resources and technology that a small start up company may not have access to.

Our strategy is Custom development. We believe that it is better to do most things ourselves. It costs less and will generally be simpler for us. Risks that we have with this strategy is that we may not be fluent in everything that we need and we may not be doing everything in the easiest way.

Evaluation criteria	Relative Importance (weight)	Alternative 1: Custom	Score (1-5)	Weighted score	Alternative 2: Outsourcing	Score (1-5)	Weighted score
Time to implement	15	Shorter if done in-house	4	60	Could take more time, waiting on product to arrive	2	30
Maintainability	10	Easy if done and reported by people in-house	5	50	Could run into issues if developers are hard to reach	3	30
ROI / long term benefit	15	Greater, no need to pay back	5	75	Less, would have to pay for software or code. Potentially need to pay them to help fix issues	3	45
Availability of resources	20	Few resources	2	40	Greater resources, more technology	5	100
Cost	10	None if done in- house	5	50	Greater cost to pay developer to create material. Potentially pay more later as well	2	20
Ease to implement	10	easy	5	50	Fairly easy if packaged well	4	40
Reliability	20	Assuming in-house developers know what they are doing, their designs should be good	4	80	Material will be created by someone who definitely knows what they are doing and should not break	4	80
	Totals:			405			345

Step 2:

TYPE: ArticleServerAPI

Method Name: GetArticles	Class Name: ArticleServerAPI	ID: 1
Contract ID: N/A	Programmer: Vinnan Muralikrishnan	Date Due: 12/3/2019
Programming Language: Go		
Triggers/Events: User clicks on a specific article or navigates to a url with an article ID to be displayed.		

Arguments Received: Data Type:	Notes:
context.Context	An object which tracks changes/updates/timeouts from the function which invoked the method.

GetArticlesRequest	gRPC Request Message containing the following fields: articleIDs: int64 - The desired article IDs to get information for
--------------------	---

Arguments Returned: Data Type:	Notes:
GetArticlesResponse	gRPC Response Message containing the following fields: articlesMap: map<int64, ArticleBasic> - A map which correlates article IDs to all of their information except the related articles
error	An error type, either signifying success if 'nil', or the error which occurred in the method.
<p>Algorithm Specification:</p> <p>For all valid article IDs provided, their information is fetched from the article DB and placed the map to be returned as part of the GetArticlesRequest.</p>	

Method Name: GetArticleFeed	Class Name: ArticleServerAPI	ID: 2
Contract ID: N/A	Programmer: Vinnan Muralikrishnan	Date Due: 12/3/2019
Programming Language: Go		
Triggers/Events: The user navigates to the main page to see the article feed for a category and location.		

Arguments Received: Data Type:	Notes:
context.Context	An object which tracks changes/updates/timeouts from the function which invoked the method.
GetArticleFeedRequest	gRPC Request Message containing the following fields: Page: int64 - the "page" the user is on relative to the number of articles already seen; 10 per page

	<p>categoryID: int64 - Optional, the category ID for the feed, if known</p> <p>feedCategory: string - the string representation of the category, used if the categoryID is not known</p> <p>locationID: int64 - Optional, the location ID for the feed, if known</p> <p>feedLocation: string - the string representation of the location, used if the locationID is not known</p> <p>lastArticleID - Optional, the article ID of the last article shown to the user in the feed, used to make sure articles returned to the user are relevant if the feed has changed</p> <p>userID: int64 - Optional, the ID of the user requesting the articles, so voting information can be populated in the response</p>
--	---

Arguments Returned:	
Data Type:	Notes:
GetArticlesResponse	<p>gRPC Response Message containing the following fields:</p> <p>feedArticles - An ordered list of articles with all information including the related articles</p> <p>maxPage: int64 - the maximum amount of pages for the feed, indicates when to stop requesting for more pages</p>

	<p>categoryID: int64 - the category ID of the feed</p> <p>categoryIsDefault: bool - whether or not the “categoryFeed” was resolved into a categoryID and the default category was provided</p> <p>locationID: int64 - the location ID of the feed</p> <p>locationIsDefault: bool - whether or not the “locationFeed” was resolved into a locationID and the default location was provided</p>
error	An error type, either signifying success if ‘nil’, or the error which occurred in the method.
<p>Algorithm Specification:</p> <p>The categoryFeed and locationFeed strings are resolved into their categoryID’s and locationID’s, respectively, or are given defaults. If either is given a default, the proper return fields categoryIsDefault and locationIsDefault will be marked true. The feed for the requested category and location will be fetched from the article DB with the page offset accounted for along with the user’s votes for each of the articles and related articles. The information is then placed in the response.</p> <p>For all valid article IDs provided, their information is fetched from the article DB and placed the map to be returned as part of the GetArticlesRequest.</p>	

Method Name: GetArticleAndRelatedArticles	Class Name: ArticleServerAPI	ID: 3
Contract ID: N/A	Programmer: Vinnan Muralikrishnan	Date Due: 12/3/2019
Programming Language: Go		
Triggers/Events: The user navigates to a page with a focussed article by accessing the article directly via url.		

Arguments Received: Data Type:	Notes:
context.Context	An object which tracks changes/updates/timeouts from the function which invoked the method.
GetArticleAndRelatedArticles Request	gRPC Request Message containing the following fields: articleID: int64 - The ID of the article that will have all of it's information along with related articles populated

	<p>userID: int64 - Optional, the ID of the user requesting the article, so voting information can be populated in the response</p>
--	--

Arguments Returned: Data Type:	Notes:
GetArticleAndRelatedArticles Response	<p>gRPC Response Message containing the following fields:</p> <p>article - An article with all information including related articles</p>
error	<p>An error type, either signifying success if 'nil', or the error which occurred in the method.</p>
<p>Algorithm Specification:</p> <p>The article with the articleID specified is fetched from the article DB and populated with the user's voting information if applicable.</p>	

Method Name: GetSearchResultsRequest	Class Name: ArticleServerAPI	ID: 4
Contract ID: N/A	Programmer: Vinnan Muralikrishnan	Date Due: 12/3/2019
Programming Language: Go		
Triggers/Events: The user types a keystroke into the article search bar.		

Arguments Received: Data Type:	Notes:
context.Context	An object which tracks changes/updates/timeouts from the function which invoked the method.
GetSearchResultsRequest	gRPC Request Message containing the following fields: query: string - The query to be searched for in the articles' titles and descriptions

	<p>step: int32 -The number of keys the user has typed since the search bar has been focussed; to synchronize the results returned</p> <p>userID: int64 - Optional, the ID of the user requesting the article, so voting information can be populated in the response</p>
--	--

Arguments Returned: Data Type:	Notes:
GetSearchResultsWithRelatedResponse	<p>gRPC Response Message containing the following fields:</p> <p>queryArticles - A list of articles matching the user's query ordered by relevance, with all information including related articles and voting information if applicable</p>
error	An error type, either signifying success if 'nil', or the error which occurred in the method.
<p>Algorithm Specification:</p> <p>The query is formatted for searching in the article DB, the executed in the article DB and populated with the user's voting information if applicable (for each returned article).</p>	

Method Name: VoteOnArticle	Class Name: ArticleServerAPI	ID: 5
Contract ID: N/A	Programmer: Vinnan Muralikrishnan	Date Due: 12/3/2019
Programming Language: Go		
Triggers/Events: The user clicks on one of the voting buttons for an article.		

Arguments Received: Data Type:	Notes:
context.Context	An object which tracks changes/updates/timeouts from the function which invoked the method.
VoteOnArticleRequest	gRPC Request Message containing the following fields:

	<p>articleID: int64 - The ID of the article that will be voted on for the user</p> <p>vote: int32 -The user's vote represented as either a -1 (downvote), 0 (neutral), or 1 (upvote).</p> <p>userID: int64 - The ID of the user requesting the article, so voting information can be populated in the response</p>
--	--

Arguments Returned: Data Type:	Notes:
VoteOnArticleResponse	<p>gRPC Response Message containing the following fields:</p> <p>Success: bool - Whether or not the vote was recorded successfully</p>
error	An error type, either signifying success if 'nil', or the error which occurred in the method.
<p>Algorithm Specification:</p> <p>The article is found in the article DB, then the user's vote is either create or updated in the article DB for that article.</p>	

TYPE: BypassServerAPI

Method Name: RemoveArticlesFromCache	Class Name: BypassServerAPI	ID: 6
Contract ID: N/A	Programmer: Vinnan Muralikrishnan	Date Due: 12/3/2019
Programming Language: Go		
Triggers/Events: ArticleServerAPI determines that an article should no longer be cached after it is deleted or disabled		

Arguments Received: Data Type:	Notes:
context.Context	An object which tracks changes/updates/timeouts from the function which invoked the method.

RemoveArticlesFromCacheRequest	gRPC Request Message containing the following fields: articleIDs: list[int64] - List of article IDs to be removed from the cache if they exist in the cache
--------------------------------	--

Argument Returned: Data Type:	Notes:
RemoveArticlesFromCacheResponse	gRPC Response Message containing the following fields: Success: bool - Whether or not the articles with the specified article IDs were purged from the cache.
error	An error type, either signifying success if 'nil', or the error which occurred in the method.
<p>Algorithm Specification:</p> <p>Each of the articles represented by the article IDs in the list provided is deleted from the Bypass database.</p>	

Method Name: AddArticlesToCache	Class Name: BypassServerAPI	ID: 7
Contract ID: N/A	Programmer: Vinnan Muralikrishnan	Date Due: 12/3/2019
Programming Language: Go		
Triggers/Events: Finder API adds articles to the Feed and therefore has a high likelihood of being requested by a user, and therefore should be prepopulated.		

Arguments Received: Data Type:	Notes:
context.Context	An object which tracks changes/updates/timeouts from the function which invoked the method.
AddArticlesToCacheRequest	gRPC Request Message containing the following fields: articleIDs: list[int64] - List of article IDs to be added to the cache if they don't already exist in the cache

Argument Returned: Data Type:	Notes:
AddArticlesToCacheResponse	gRPC Response Message containing the following fields: Success: bool - Whether or not the articles with the specified article IDs were added to the cache.
error	An error type, either signifying success if 'nil', or the error which occurred in the method.
<p>Algorithm Specification:</p> <p>Each of the articles represented by the article IDs in the list provided that do not exist in the Bypass database have the HTML at their URL fetched and groomed for the user to see in an iFrame, then inserted into the Bypass database.</p>	

Method Name: CheckArticleExistence	Class Name: BypassServerAPI	ID: 8
---------------------------------------	--------------------------------	-------

Contract ID: N/A	Programmer: Vinnan Muralikrishnan	Date Due: 12/3/2019
Programming Language: Go		
Triggers/Events: The user clicks on an article and wants to see the HTML for an article.		

Arguments Received: Data Type:	Notes:
context.Context	An object which tracks changes/updates/timeouts from the function which invoked the method.
CheckArticleExistenceRequest	gRPC Request Message containing the following fields: articleID: int64 - The article ID to check the existence of

Argument Returned: Data Type:	Notes:
CheckArticleExistenceResponse	gRPC Response Message containing the following fields: exists: bool - Whether or not the articles with the specified article ID exists in the cache.
error	An error type, either signifying success if 'nil', or the error which occurred in the method.
<p>Algorithm Specification:</p> <p>The articles represented by the article ID is checked for existence in the Bypass DB, and returned in the response if it exists. Otherwise, it's HTML is fetched and groomed, then a successful existence is returned and the fetched HTML is inserted into the Bypass database.</p>	

Method Name: GetArticleCache	Class Name: BypassServerAPI	ID: 9
Contract ID: N/A	Programmer: Vinnan Muralikrishnan	Date Due: 12/3/2019

Programming Language: Go

Triggers/Events: The user has already verified the article exists in the cache and is now requesting the actual HTML.

Arguments Received: Data Type:	Notes:
context.Context	An object which tracks changes/updates/timeouts from the function which invoked the method.
GetArticleCacheRequest	gRPC Request Message containing the following fields: articleID: int64 - The article ID to get the HTML for

Argument Returned: Data Type:	Notes:
	gRPC Response Message containing the following fields:

GetArticleCacheResponse	articleHTML: List[byte] - The HTML for the article with the given article ID
error	An error type, either signifying success if 'nil', or the error which occurred in the method.
<p>Algorithm Specification:</p> <p>The articles represented by the article ID is checked for existence in the Bypass DB, and returned in the response if it exists. Otherwise it's HTML is fetched and groomed, then a successful existence is returned and the fetched HTML is inserted into the Bypass database.</p>	

Method Name: GetArticleCache	Class Name: BypassServerAPI	ID: 10
Contract ID: N/A	Programmer: Vinnan Muralikrishnan	Date Due: 12/3/2019
Programming Language: Go		

Triggers/Events: The user has already requested the article HTML and further assets for the article need to be recieved.

Arguments Received: Data Type:	Notes:
context.Context	An object which tracks changes/updates/timeouts from the function which invoked the method.
ResolveArticleRequest	gRPC Request Message containing the following fields: articleID: int64 - The article ID to get the URL for

Argument Returned: Data Type:	Notes:
ResolveArticleResponse	gRPC Response Message containing the following fields: url: string - The URL for the article with the given article ID

error	An error type, either signifying success if 'nil', or the error which occurred in the method.
<p>Algorithm Specification:</p> <p>The article is found in the Bypass DB, and it's URL is returned in the response.</p>	

TYPE: AuthServerAPI

Method Name: TokenValid	Class Name: AuthServerAPI	ID: 11
Contract ID: N/A	Programmer: Vinnan Muralikrishnan	Date Due: 12/3/2019
Programming Language: Go		
Triggers/Events: The user requests functionality from another back-end API.		

Arguments Received: Data Type:	Notes:
context.Context	An object which tracks changes/updates/timeouts from the function which invoked the method.
TokenValidRequest	gRPC Request Message containing the following fields: token: string - The auth token for the user userLevel: int32 - The auth level for the requested action

Argument Returned: Data Type:	Notes:
TokenValidResponse	gRPC Response Message containing the following fields: hasAccess: bool - Whether or not the user has access to the requested functionality logout: bool - Whether or not the user's token has expired and they need to be logged out userID: int64 - The userID belonging to the user with the provided token
error	An error type, either signifying success if 'nil', or the error which occurred in the method.

Algorithm Specification: The token information is resolved by fetching from the Auth DB and then checking the constraints in the request against the user's userLevel and session duration.

Method Name: AuthenticateUser	Class Name: AuthServerAPI	ID: 12
Contract ID: N/A	Programmer: Vinnan Muralikrishnan	Date Due: 12/3/2019
Programming Language: Go		
Triggers/Events: The user logs in via another service (Facebook, Twitter, Microsoft, etc)		

Arguments Received: Data Type:	Notes:
---------------------------------------	--------

context.Context	An object which tracks changes/updates/timeouts from the function which invoked the method.
AuthenticateUserRequest	gRPC Request Message containing the following fields: email: string - The email of the user logging in

Argument Returned: Data Type:	Notes:
AuthenticateUserResponse	gRPC Response Message containing the following fields: token: string - The auth token for the user Expires: int64 - Timestamp of expiration date of token
error	An error type, either signifying success if 'nil', or the error which occurred in the method.
<p>Algorithm Specification: If the user exists, they are given a new auth token and session expiration time, and that session is created in the Auth DB. Otherwise, the user is created before sending back the session token.</p>	

Method Name: TokenInvalidate	Class Name: AuthServerAPI	ID: 13
Contract ID: N/A	Programmer: Vinnan Muralikrishnan	Date Due: 12/3/2019
Programming Language: Go		
Triggers/Events: The user logs out or requests to log out another open session of theirs.		

Arguments Received: Data Type:	Notes:
context.Context	An object which tracks changes/updates/timeouts from the function which invoked the method.
TokenInvalidateRequest	gRPC Request Message containing the following fields: currentToken: string - The current token of the user invalidateToken: string - The token the user wants to invalidate

Argument Returned: Data Type:	Notes:
TokenInvalidateResponse	gRPC Response Message containing the following fields: success: bool - Whether or not the user was successful in the operation
error	An error type, either signifying success if 'nil', or the error which occurred in the method.
Algorithm Specification: If the user owns the token to invalidate, it is invalidated in the Auth DB, otherwise it is ignored, the request is logged, and an unsuccessful response is returned.	

Method Name: UpdateUserSessionExpiration	Class Name: AuthServerAPI	ID: 14
Contract ID: N/A	Programmer: Vinnan Muralikrishnan	Date Due: 12/3/2019

Programming Language: Go		
Triggers/Events: The user wants to change when their sessions expire.		

Arguments Received: Data Type:	Notes:
context.Context	An object which tracks changes/updates/timeouts from the function which invoked the method.
UpdateUserSessionExpiration Request	gRPC Request Message containing the following fields: userID: int64 - The userID belonging to the user newUserSessionExpiration: int64 - The number of minutes the user wants their logged in sessions to last before they need to reauthenticate

Argument Returned: Data Type:	Notes:
UpdateUserResponse	gRPC Response Message containing the following fields: success: bool - Whether or not the user was successful in the operation
error	An error type, either signifying success if 'nil', or the error which occurred in the method.
Algorithm Specification: The user's session expiration time is updated in the Auth DB.	

Method Name: CheckUserPremium	Class Name: AuthServerAPI	ID: 15
Contract ID: N/A	Programmer: Vinnan Muralikrishnan	Date Due: 12/3/2019
Programming Language: Go		

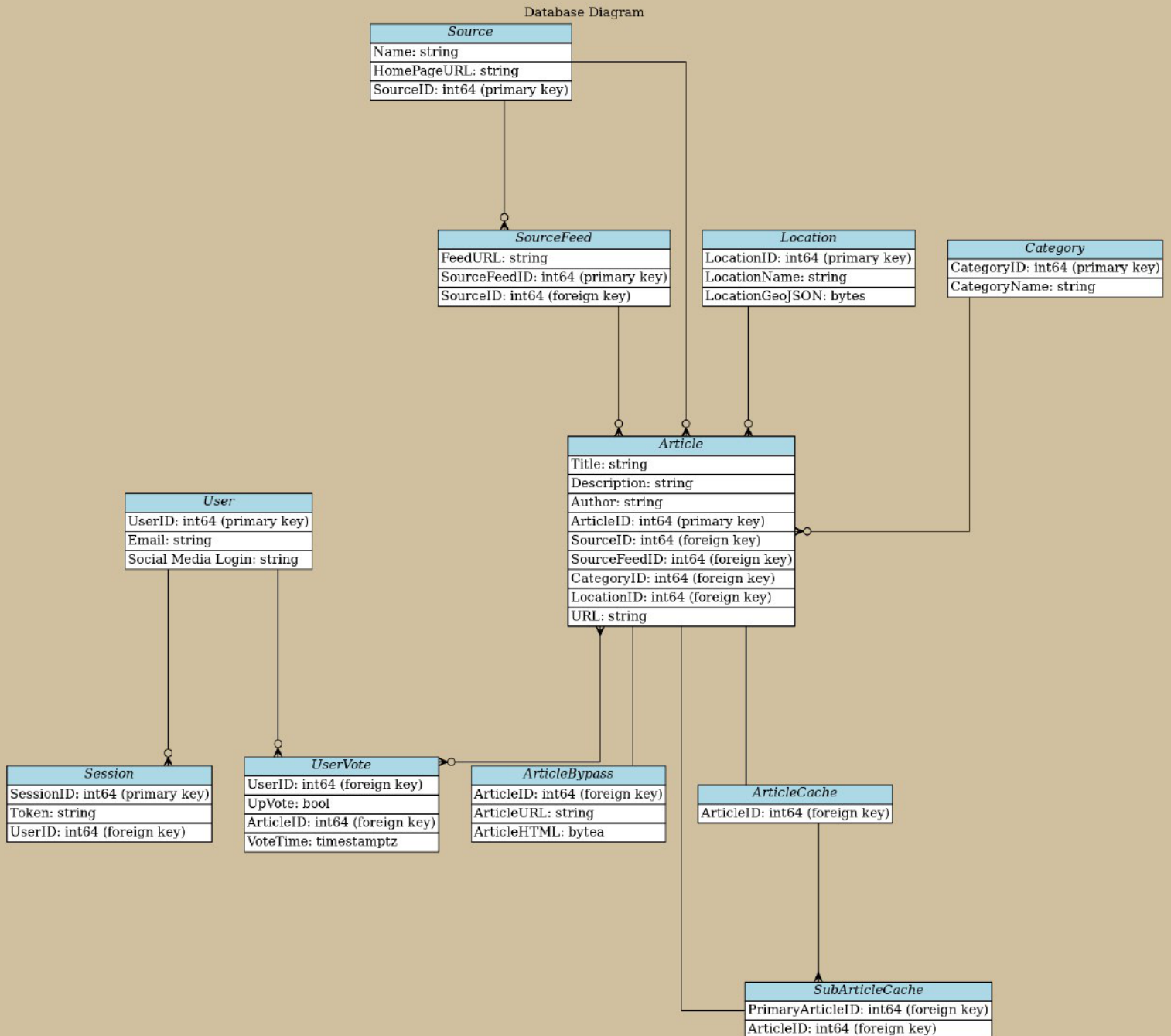
Triggers/Events: The user requests an operation only available to premium tier users and a back-end service needs to check whether or not they have access.

Arguments Received: Data Type:	Notes:
context.Context	An object which tracks changes/updates/timeouts from the function which invoked the method.
CheckUserPremiumRequest	gRPC Request Message containing the following fields: userID: int64 - The userID belonging to the user

Argument Returned: Data Type:	Notes:
CheckUserPremiumResponse	gRPC Response Message containing the following fields: premium: bool - Whether or not the user is premium

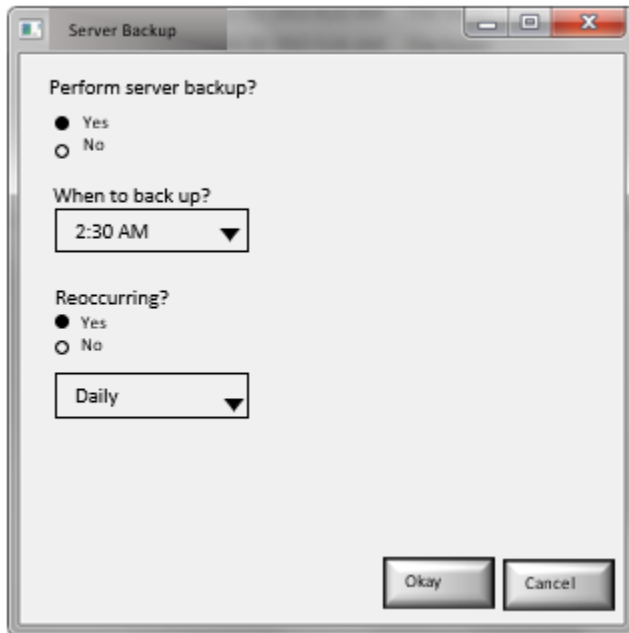
error	An error type, either signifying success if 'nil', or the error which occurred in the method.
Algorithm Specification: The user's premium status is fetched from the auth DB and returned.	

Step 3:



Step 4:

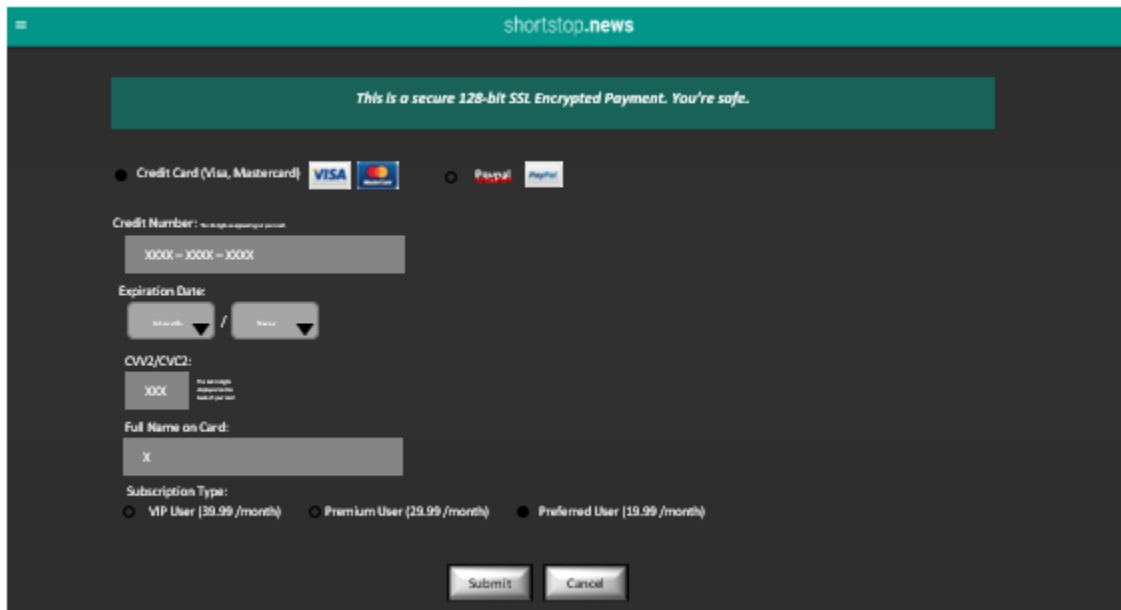
Use Case 1: Auto Backups



A Windows-style dialog box titled "Server Backup". It contains the following elements:

- Perform server backup?**
 - ☒ Yes
 - ☐ No
- When to back up?**
 - A dropdown menu showing "2:30 AM".
- Reoccurring?**
 - ☒ Yes
 - ☐ No
- Frequency:**
 - A dropdown menu showing "Daily".
- Buttons:** "Okay" and "Cancel" at the bottom right.

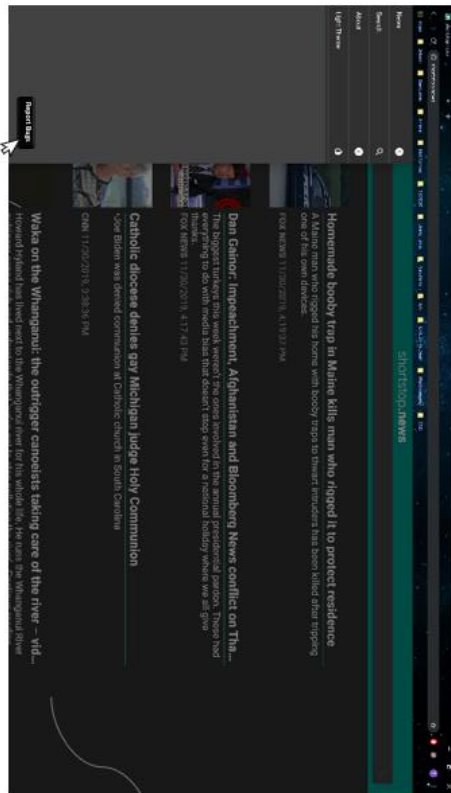
Use Case 2: Secure Payments



A payment form for "shortstop.news" with a teal header. It includes a security message and various input fields:

- Security Message:** "This is a secure 128-bit SSL Encrypted Payment. You're safe."
- Payment Method:** Radio buttons for "Credit Card (Visa, Mastercard)" (selected) and "Paypal". Logos for Visa, Mastercard, and PayPal are shown.
- Credit Number:** A text field with placeholder "XXXX - XXXX - XXXX".
- Expiration Date:** Two dropdown menus for month and year, separated by a slash.
- CVV2/CVC2:** A text field with placeholder "XXX".
- Full Name on Card:** A text field with placeholder "X".
- Subscription Type:** Radio buttons for "VIP User (\$9.99 /month)", "Premium User (\$9.99 /month)", and "Preferred User (\$9.99 /month)".
- Buttons:** "Submit" and "Cancel" at the bottom.

Use Case 3: Multi User Support (More specifically the “Report Bugs” button)



Bug Report

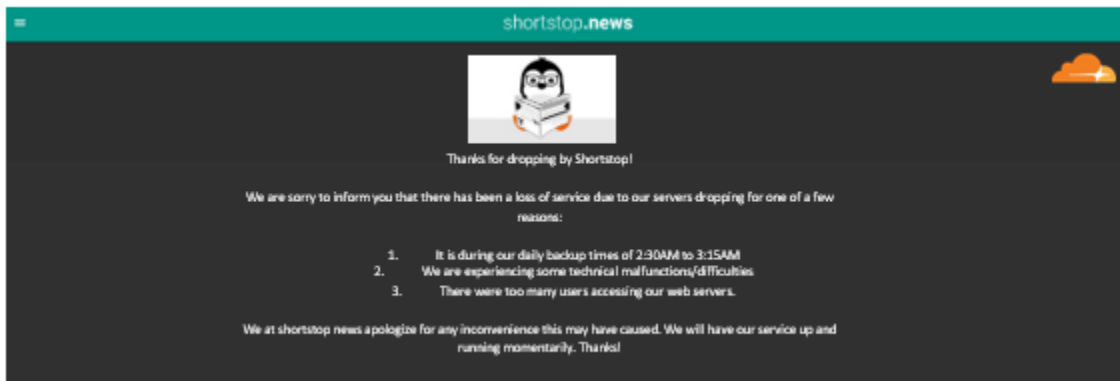
Please describe the bug:

2:30 AM – The website crashed and wouldn't let me view my news! I need the news. It's so boring in this little town. I need to read about homemade booby traps killing the person who rigged it. Or about Impeachment. Why did your website crash?!

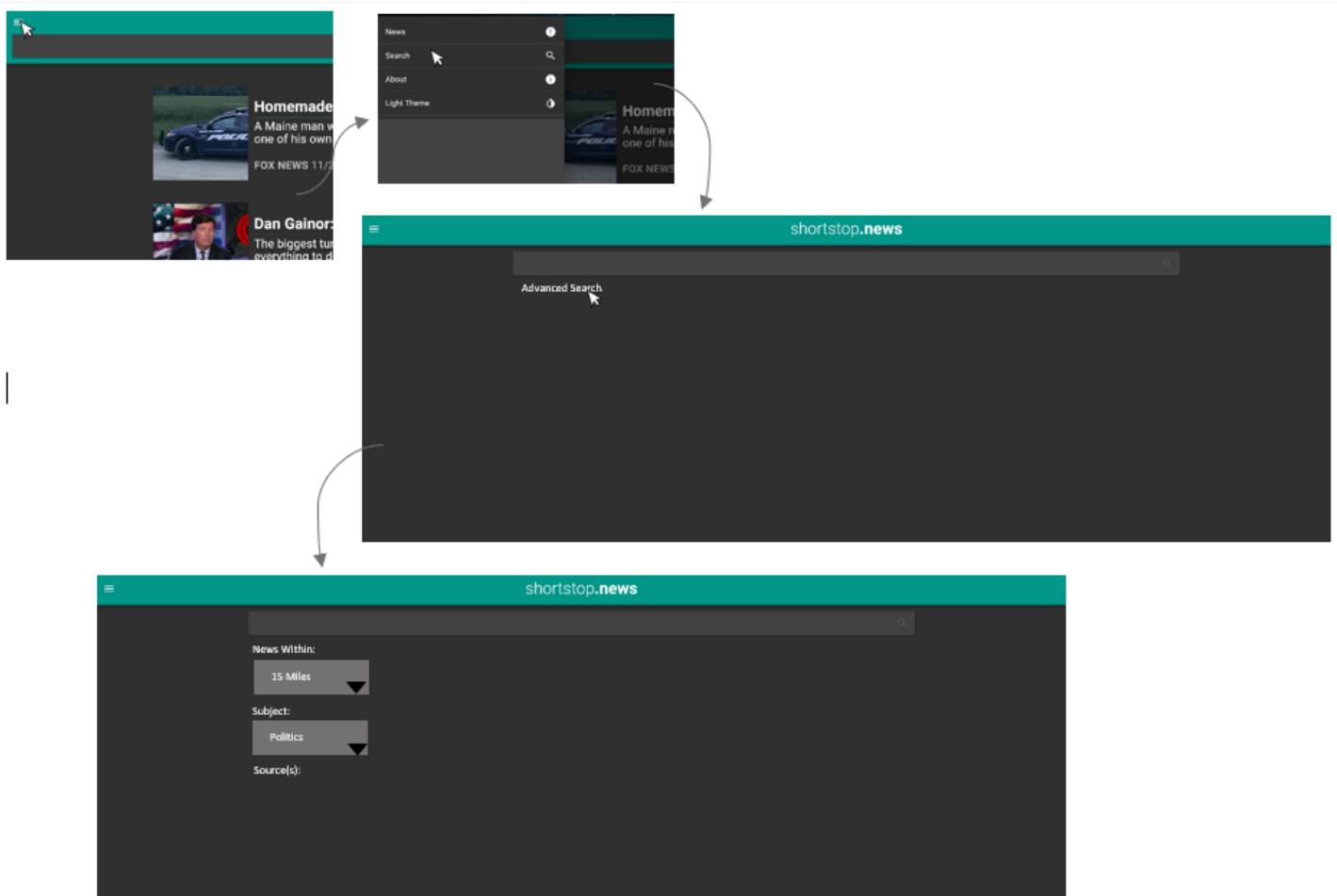
Characters Remaining: 7 of 250 (not used)

Submit **Cancel**

Use Case 4: Run 24/7 (In the case of service interruption):



Use Case 5: Friendly Website



Step 5:

Software and Hardware Specification

Operating System:

Our standard client can use Windows, Internet Explorer, Mozilla Firefox, Microsoft Edge or any other popular web browsers. The Web Server ,the Application Server, Database Server we have is Linux.

Special Software:

Our web server uses apache HTTP Server, our application server uses Java Server Pages, our database server uses Oracle database.

Hardware:

Our standard client does not need a lot of memory and disk drive as ours is a web-based news feed. Our web server needs 16 GB memory and 1TB disk drive, our application server needs 32GB memory and 2TB disk drive and database needs 64GB memory and 4TB disk drive.

Our standard client can use Intel Core, AMD Ryzen, or any other popular processor. Our web server uses Intel Xenon and our application server, database server uses Intel Xenon E5-2600.

Network:

Our standard client needs a minimum of 50 Mbps for optimum browsing. Our web server, application server, database server uses 100 Mbps Ethernet.

Safety requirements:

If there is massive damage to the majority portion of the database due to some catastrophic failure, such as a disk crash, we can restore our database. We do it by using the past database copy that was backed up to archival storage and reconstruct a current state by reapplying the operations of committed transactions from the backed-up log, until the time of failure.

Security requirements:

We place a huge emphasis on security. We get cash payment through PayPal and top credit/debit cards only. We have our database server under tight physical security. We ask users to confirm that they are not a robot when they need to ask any queries. Our U.S.-based customer service team is available 24/7 365 to assist with security issues and questions.